1

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 10-198563
(43)Date of publication of application : 31.07.1998

(51)Int.Cl. G06F 9/38

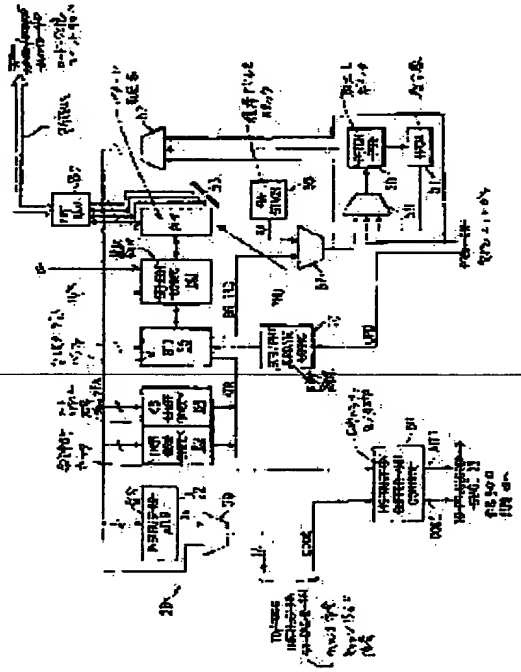
(21)Application number : 10-031921
(22)Date of filing : 05.01.1998

(71)Applicant : TEXAS INSTR INC <TI>
(72)Inventor : JOHNATHAN H SHELL
GEORGE Z N KAI

(30)Priority
Priority number : 96 34397 Priority date : 30.12.1996 Priority country : US

(54) DYNAMICALLY LOADABLE PATTERN HISTORY CHART IN MICROPROCESSOR

(57)Abstract:
PROBLEM TO BE SOLVED: To provide a method for dynamically predicting the branch instruction of a microprocessor operating under a multitask environment by holding predicted information based on its own branch pattern history.
SOLUTION: A fetching unit 26 is branched with a branch target buffer 56 and plural pattern history charts 53. A selection logic 80 receives a signal indicating the kind of a program including an instruction for each branch instruction, selects one pattern history chart 53, and generates a prediction code according to a branch history field BH in the entry of the branch target buffer 56 corresponding to an instruction address by using the pattern history chart. At the time of task switching, the contents of more than one pattern history charts 53 are stored in a task state segment corresponding to an interrupted task, and an entry from the task state segment of a new task is loaded to the patter history chart 53.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

引用文献1

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平10－198563

(43)公開日　平成10年(1998)７月31日

(51)Int.Cl.[6]	識別記号	ＦＩ
Ｇ０６Ｆ　9/38	３３０	Ｇ０６Ｆ　9/38　３３０Ｂ

審査請求　未請求　請求項の数２　書面　（全 20 頁）

(21)出願番号　特願平10－31921

(22)出願日　平成10年(1998)１月５日

(31)優先権主張番号　０３４３９７

(32)優先日　1996年12月30日

(33)優先権主張国　米国（ＵＳ）

(71)出願人　590000879
テキサス　インスツルメンツ　インコーポレイテッド
アメリカ合衆国テキサス州ダラス，ノース　セントラルエクスプレスウエイ　13500

(72)発明者　ジョナサン　エィチ．シエル
アメリカ合衆国テキサス州プラノ，ロングフェロー　ドライブ　4300

(72)発明者　ジョージ　ゼット．エヌ．カイ
アメリカ合衆国テキサス州プラノ，クラレンドン　ドライブ　5720

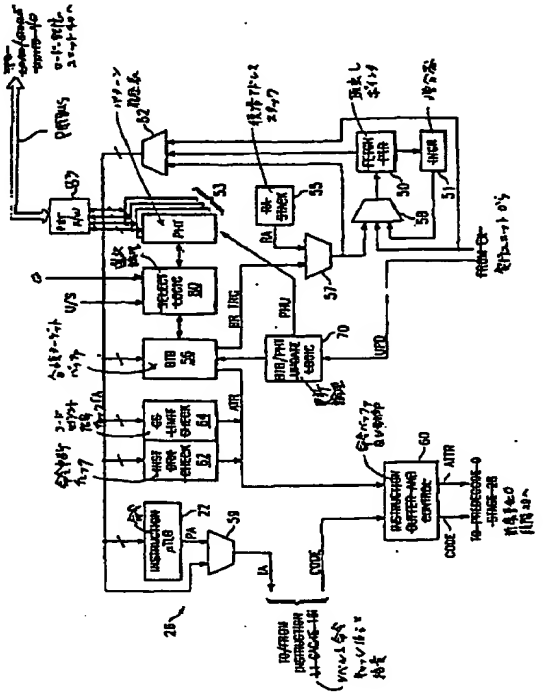
(74)代理人　弁理士　浅村　皓　（外３名）

(54)【発明の名称】　マイクロプロセッサ内の動的にロード可能なパターン履歴表

(57)【要約】

【課題】マルチタスク環境で動作するマイクロプロセッサの分岐命令を、自身の分岐パターン履歴に基づく予測情報を保持して動的に予測する方法を提供する。

【解決手段】取出しユニット２６は分岐ターゲットバッファ５６と、複数のパターン履歴表５３を有する。選択論理８０は各分岐命令毎に、命令を含むプログラムの種類を示す信号を受けて１つのパターン履歴表５３を選択し、これを用いて、命令アドレスに対応する、分岐ターゲットバッファ５６のエントリ内の分岐履歴フィールドＢＨに応じて、予測コードを生成する。タスク切替えの場合は、１つ以上のパターン履歴表５３の内容を、中断されたタスクに対応するタスク状態セグメント９０に記憶し、新しいタスクのタスク状態セグメントからのエントリをパターン履歴表５３にロードする。



【特許請求の範囲】
【請求項１】多重タスクモードで動作するマイクロプロセッサであって、
第１及び第２のタスクに従って命令を実行する少なくとも１つの実行ユニットと、
前記第１及び第２タスクのそれぞれに関連する部分を含み、また命令を記憶する部分を含む、メモリと、
メモリにアドレスして前記実行ユニットが実行する命令コードを検索する取出しユニットであって、
前記実行ユニットが実行した分岐命令の一連の結果を記憶する分岐履歴回路と、
前記分岐履歴回路に結合し、前記分岐履歴回路からの分岐履歴フィールドに対応する予測情報を与える、パターン履歴回路と、
取り出す命令のアドレスを選択する、アドレス指定回路と、を備える取出しユニットと、
前記パターン履歴回路と前記メモリに結合し、前記第１タスクから前記第２タスクへのタスク切替えに応じて前記予測情報を修正する、回路と、を備える、マイクロプロセッサ。
【請求項２】パイプライン化マルチタスクマイクロプロセッサを操作する方法であって、
パイプライン化マイクロプロセッサの取出し段階で第１のタスクの分岐命令を検出し、
前記検出ステップに応じて、分岐履歴フィールドの少なくとも一部を検索し、
前記分岐履歴フィールドの前記検索された部分に対応する記憶された予測情報から、分岐予測を生成し、
前記第１タスクから第２のタスクへのタスク切替えに応じて、前記予測情報を修正する、パイプライン化マルチタスクマイクロプロセッサを操作する方法。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】この発明はマイクロプロセッサの分野に関し、より特定すると、パイプライン化マイクロプロセッサにおける分岐予測法に関する。
【０００２】
【従来の技術】マイクロプロセッサやその他のプログラマブル論理デバイスの分野では近年多くの改善が行われ、性能が飛躍的に改善された。その一例はパイプライン化アーキテクチャであって、多数のマイクロプロセッサ命令を種々の実行段階で同時に処理するので、前の命令が完了する前に次の命令の処理が始まる。個々の命令を処理するのに取出しから実行まで数マシンサイクルを要するにも関わらず、パイプライン方式を用いると、単一パイプラインマイクロプロセッサ内でマイクロプロセッサが命令を実行する有効速度は、マシンサイクル当たり１命令に近くなる。いわゆるスーパースカラアーキテクチャでは多数のパイプラインが並列に動作するので、理論的な性能レベルは更に高い。
【０００３】この技術で知られているように、多くの従来のコンピュータ及びマイクロプロセッサのプログラムでは分岐命令が用いられる。分岐命令はプログラムの流れを変える命令であって、分岐命令の後に実行する次の命令は、必ずしもプログラムの順序における次の命令ではない。分岐命令には、ＪＵＭＰ命令やサブルーチン呼出しやサブルーチン復帰などの無条件分岐命令と、前の論理や算術命令の結果に依存する条件付き分岐命令がある。
【０００４】条件付き分岐命令があるとマイクロプロセッサのパイプライン化アーキテクチャは複雑になる。それは、分岐する条件は実行するまで分からず、実行は取出してから数サイクル後になるからである。この状況では、マイクロプロセッサは分岐条件が分かるまで分岐後の命令の取出しを中止し、その結果パイプラインに空虚な段階である「バブル」（即ち、命令処理の穴）を生じるか、またはパイプラインを満たすために命令を推測により（実際には条件を憶測して）取り出して、もし推測が間違ったことが分かると現在の命令のパイプラインを「フラッシュ」するという危険を冒すか、のどちらかである。
【０００５】命令を推測により実行してパイプラインを常に満たすことの利点は、特に長いまたは多数のパイプラインアーキテクチャでは、推測による実行の成功率が所望の性能を得るのに十分である限り、パイプラインのフラッシュによる性能の低下を十分補うことである。したがって現在の多くのマイクロプロセッサは、条件付き分岐命令の行動をある程度の正確さで予測できる何らかの分岐予測法を用いている。分岐予測の種類の１つは、予測が時間や履歴に従って変わらない「静的」予測である。簡単な静的予測法は、単に全ての条件付き分岐を「行う」として予測する。より進んだ静的分岐予測法は分岐の方向に従って予測する。例えば全ての順方向の条件付き分岐を「行わない」と予測し、全ての逆方向の分岐（例えば、ＤＯループ内のＬＯＯＰ命令）を「行う」と予測する。もちろん、無条件分岐の場合は必ず静的に「行う」と予測する。
【０００６】動的分岐予測は、過去の分岐の結果を用いて次の分岐の結果を予測する、既に知られている分岐予測法である。よく知られた簡単な動的予測法は、単に最近の１つまたは２つの条件付き分岐の結果を用いて現在の分岐命令の方向を予測する。
【０００７】より正確な動的な分岐予測法は、他の命令の分岐結果ではなく自分の分岐履歴を用いて分岐命令の方向を予測する。この方法は最近のマイクロプロセッサでは、一般的に分岐ターゲットバッファを用いて行われている。従来の分岐ターゲットバッファＢＴＢはキャッシュのエントリに似た表で、各エントリは最近出現した分岐命令の識別子（「タグ」）と、予測を行うための分岐履歴に関するコードと、分岐を行うと予測した場合に

取り出す次の命令のターゲットアドレス（その次の順次のアドレスは「行わない」予測で取り出すアドレス）を記憶する。分岐命令を取り出すと、そのアドレスをＢＴＢ内のタグと比べて、この命令が前に出現したかどうか判断する。出現した場合は、その命令についてＢＴＢ内で示された予測コードに従って次の命令を取り出す。新しく出現した分岐命令については履歴がＢＴＢ内にないので、静的に予測する。命令を実行して完了すると、ＢＴＢエントリを作り（一般に、分岐を行ったものについてだけ）または修正して（すでにＢＴＢエントリを持つ分岐について）分岐命令の実際の結果を反映し、これを次にその命令が起こったときに用いる。

【０００８】最も近くに実行した分岐かまたは同じ命令の分岐履歴に基づいて分岐を予測する種々の実際の予測アルゴリズムが知られている。よく知られた簡単な分岐アルゴリズムは４状態の状態機械モデルに従い、最も近い２つの分岐事象を用いて次の分岐を行うか行わないかを予測する。４状態とは、「強く行う」と、「行う」と、「行わない」と、「強く行わない」である。「強く」の状態は、少なくとも最後の２つの分岐が全て「行う」だけか「行わない」だけの場合（実現したものに従って、一般に、またはその特定の命令について）に対応する。「行う」状態と「行わない」状態（即ち「強く」の状態でない）は最後の２つの分岐の結果が異なる場合に対応し、次の分岐は、予測を逆の方向に変えるか、または予測の方向を保ちしかも「強く」にする。

【０００９】最近の進んだ分岐予測アルゴリズムは、分岐行動を予測するのに、分岐履歴だけでなく分岐パターン情報を用いる。例えば或る分岐命令は３回通るループで、その分岐履歴は行う・行う・行わない、というパターンを繰り返す。簡単な２ビット即ち４状態の予測法を用いたのでは、その行動が完全に予測可能であってもこの命令の分岐を正しくは予測しない。よく知られた２レベル適応分岐予測法が、Ｙｅｈ　とｐａｔｔの「２レベル適応分岐予測（Ｔｗｏ－Ｌｅｖｅｌ　Ａｄａｐｔｉｖｅ　Ｂｒａｎｃｈ　Ｐｒｅｄｉｃｔｉｏｎ）」、マイクロアーキテクチャに関する第２４回国際シンポジウム議事録（Ｐｒｏｃｅｅｄｉｎｇｓ　ｏｆｔｈｅ　２４ｔｈ　Ｉｎｔｅｒｎａｔｉｏｎａｌ　Ｓｙｍｐｏｓｉｕｍ　ｏｎＭｉｃｒｏａｒｃｈｉｔｅｃｔｕｒｅ）（ＡＣＭ／ＩＥＥＥ、１９９１年１１月）、５１－６１ページ、に述べられている。この方法は、分岐履歴と共に分岐パターン情報を用いて分岐命令の結果を予測する。Ｙｅｈとｐａｔｔの方法を用いた分岐予測がＢＴＢを用いるマイクロプロセッサアーキテクチャに適用され、英国特許出願番号第２　２８５　５２６号、１９９５年７月１２日発行に述べられている。またこれについては、米国特許番号第５，５７４，８７１号を参照のこと。

【００１０】上記のＹｅｈ　とｐａｔｔの論文と英国特許出願番号第２　２８５　５２６号に述べられている方法では、特有の分岐パターン毎にパターン履歴を保持し、また更新する。この方法では、パターン履歴は上述の４状態の状態機械モデルから成り、分岐パターン毎に最近の２つの分岐事象を用いて、同じ分岐パターンを持つ次に発生する分岐が「行う」か「行わない」かを予測する。（その「強く」という属性と共に）。動作を説明すると、ＢＴＢ内にエントリを持つ分岐命令を検出すると、その命令の分岐履歴フィールドに含まれる分岐パターンを用いてパターン履歴表に指標を付け、そこから予測を取り出す。分岐が決定すると、その特定の命令の分岐履歴フィールドと前のパターンのパターン履歴（即ち、予測に用いた分岐パターン）を更新する。この更新されたパターン履歴を用いて、その関連する分岐パターンをＢＴＢの分岐履歴フィールド内に持つ次の分岐命令の結果を予測する。したがって、この方法によるパターン履歴表は「グローバル」である。その意味は、命令が何かに関係なく、同じ分岐履歴パターンを持つ任意の分岐命令について分岐予測を生成する、ということである。したがって或る特定の分岐パターンについてのパターン履歴は、その分岐履歴を持つ任意の分岐命令の分岐予測の結果に基づいて定義し更新する。このようこの基本的な２レベル法では、別の命令の分岐結果に基づいて、任意の所定の命令の分岐予測を決定する。

【００１１】ＹｅｈとＰａｔｔの「２レベル適応分岐予測の別の方法（Ａｌｔｅｒｎａｔｉｖｅ　Ｉｍｐｌｅｍｅｎｔａｔｉｏｎｓ　ｏｆ　Ｔｗｏ－Ｌｅｖｅｌ　Ａｄａｐｔｉｖｅ　Ｂｒａｎｃｈ　Ｐｒｅｄｉｃｔｉｏｎ）」、コンピュータアーキテクチャに関する第１９回年次国際シンポジウム議事録（Ｃｏｎｆｅｒｅｎｃｅ　Ｐｒｏｃｅｅｄｉｎｇｓ　ｏｆ　ｔｈｅ　１９ｔｈ　Ａｎｎｕａｌ　Ｉｎｔｅｒｎａｔｉｏｎａｌ　Ｓｙｍｐｏｓｉｕｍ　ｏｎ　Ｃｏｍｐｕｔｅｒ　Ａｒｃｈｉｔｅｃｔｕｒｅ）、（ＡＣＭ、１９９２年５月）、１２４－１３４ページ、に述べられているように、２レベル分岐予測の別の方法はこの制限に対処する。この論文の図３に示すように、この別の方法はアドレス特有のパターン履歴表を作り、ＢＴＢ内の各エントリは独自のパターン履歴表を持つ。したがって、分岐命令の分岐予測は、自身の過去の履歴から生成し修正したパターン履歴に基づいて作り、同様の分岐パターンを持つ他の分岐命令の分岐結果は用いない。

【００１２】

【発明が解決しようとする課題】アドレス特有のパターン履歴表を用いると同じ分岐パターンを持つ他の分岐命令から得た分岐予測に含まれる干渉はなくなるが、これを実現するコストは非常に大きい。例えば、現在のマイクロプロセッサが持つＢＴＢは４ｋエントリにもなる。したがってアドレス特有のパターン履歴表に４ビットの分岐履歴の指標を用いると、それぞれが２ビット幅の１６エントリを持つ４ｋのパターン履歴表が必要で、記憶

量は１２８ｋビットになる。したがってこの方法を実現するのに必要なチップ面積は非常に大きい。しかも、パターン履歴表の指標に分岐履歴ビットを追加して分岐予測を改善しようとすると、このコストは急速に増える。例えば、分岐履歴に６ビットを用いると５１２ｋビットのパターン履歴の記憶が必要になる。それぞれの段階が一層深い多くのパイプラインをマイクロプロセッサが持つに従って、分岐の予測誤りによる損失はますます大きくなり、正確な分岐予測にかかるプレミアムは更に高くなり、アドレス特有のパターン履歴表を実現するコストは一層大きくなる。

【００１３】更に別の背景として、種類が異なるマイクロプロセッサプログラムの分岐行動は、種類が同じであれば似ているが、別の種類の間では異なることが分かった。例えば、Ｃａｌｄｅｒ　と　Ｇｒｕｎｗａｌｄの「ライブラリ内の分岐の予測性（Ｔｈｅ　Ｐｒｅｄｉｃｔａｂｌｉｔｙ　ｏｆ　Ｂｒａｎｃｈｅｓ　ｉｎ　Ｌｉｂｒａｒｉｅｓ）」、マイクロアーキテクチャに関する第２８回国際シンポジウム議事録（ＡＣＭ／ＩＥＥＥ、１９９５年１１月）、２４－３４ページ、に述べられているように、普通使われているＵＮＩＸライブラリのサブルーチンは予測可能な分岐行動をとり、クラス即ち種類としては非ライブラリプログラムとは異なる分岐行動をとる。

【００１４】更に背景として、分岐履歴とＢＴＢのタグフィールドの一部を用いてグローバルパターン履歴表に指標を付ける方法が知られている。

【００１５】また別の背景として、現在のマイクロプロセッサはマルチタスクオペレーティングシステムを支援するようになり、マイクロプロセッサは複数のタスクの間で動作を順次に切り替えて、あたかも多数のタスクを並列に実行しているように見える。一般に、例えばよく知られたｘ８６アーキテクチャで作られたマイクロプロセッサでは、各タスクを短時間実行し、次にタスク切替えと呼ぶ事象で中断した後、別のタスクを開始または再開し、これを短時間実行し、また別のタスクに切り替えるということを続ける。このようなマルチタスク動作を行うには、各タスクのシステム文脈をタスクの中断のときに保存し、タスクの再開のときに復元しなければならない。一般にはメモリの一部を確保し、これを各タスクのシステム文脈の記憶と呼出しに用いる。ｘ８６アーキテクチャでは、タスク状態セグメント（ＴＳＳ）と呼ぶシステムセグメントを各タスクに割り当てて、タスク切替えによる中断のときにその条件を記憶する。

【００１６】

【課題を解決するための手段】したがってこの発明の目的は、プログラムの種類に基づくマイクロプロセッサ内の分岐予測を与えることである。

【００１７】この発明の別の目的は、分岐パターン履歴表をタスク毎に保持して、ある活動的なタスク内の分岐活動が、中断されたタスクの分岐パターン履歴に影響を与えないようにする、分岐予測を与えることである。この発明のその他の目的や利点は、図面と共にこの明細書を参照すれば当業者に明らかである。

【００１８】この発明は、マルチタスクのマイクロプロセッサに１つ以上の動的に書換え可能なパターン履歴表を与えることにより、マイクロプロセッサで実現することができる。マイクロプロセッサが実行する各タスクは中断されたタスクの条件を記憶する状態セグメントをメモリ内に保持し、タスクを再開するときにその動作の条件を検索する。この発明は、タスク切替えのときに、中断されるタスクのタスク状態セグメント内のパターン履歴表の内容を記憶することにより実現される。タスクを再開するタスク切替えのときに、タスク状態セグメントに記憶した内容でパターン履歴表を書き換える。

【００１９】

【発明の実施の形態】図１は、この発明の好ましい実施の形態を実現する例示のスーパースカラ・パイプライン・マイクロプロセッサ１０を備える、例示のデータ処理システム３００を示す。システム３００とマイクロプロセッサ１０のアーキテクチャは一例であって、この発明は種々のアーキテクチャのマイクロプロセッサに用いてよいものである。したがって当業者はこの明細書を参照して、他のマイクロプロセッサアーキテクチャでこの発明を容易に実現することができる。更に、この発明は単一チップマイクロプロセッサやマイクロコンピュータや多チップで実現してよく、これらの集積回路の製造は、シリコン基板、シリコン・オン・インシュレータ、ガリウム砒素、その他の製造技術により、またＭＯＳ、ＣＭＯＳ、バーポーラ、ＢｉＣＭＯＳ、その他のデバイスを用いて実現することができる。

【００２０】図１に示すように、マイクロプロセッサ１０は外部バスＢＵＳにより他のシステムデバイスに接続する。この例ではバスＢＵＳを単一バスで示しているが、ＰＣＩローカルバスアーキテクチャを用いる従来のコンピュータで知られているように、外部バスＢＵＳは異なる速度とプロトコルを持つ多重バスを表してよいことは言うまでもない。システム３００は、次の従来のサブシステムを備える。即ち、通信ポート３０３（モデムポート及びモデム、網インターフェースなどを含む）、グラフィックディスプレイ装置３０４（ビデオメモリ、ビデオプロセッサ、グラフィックモニタを含む）、一般にダイナミック・ランダムアクセスメモリ（ＤＲＡＭ）で実現されまたメモリスタック３０７を含む主メモリサブシステム３０５、入力デバイス３０６（キーボード、位置決め装置、そのインターフェース回路を含む）、ディスク装置３０８（ハードディスクドライブ、フロッピディスクドライブ、ＣＤ　ＲＯＭドライブを含む）などである。したがって図１のシステム３００は、現在では普通になっている従来のデスクトップコンピュータやワ

ークステーションに対応すると考えてよい。当業者が認めるように、マイクロプロセッサ１０の他のシステム構成もこの発明を有効に利用することができる。

【００２１】マイクロプロセッサ１０はバスインターフェースユニット（ＢＩＵ）１２を備える。ＢＩＵ１２は外部バスＢＵＳに接続し、マイクロプロセッサ１０とシステム３００内の他の構成要素との間の通信を制御し実行する。ＢＩＵ１２はこの機能を実行するための制御及びクロック回路を備える。例えば、動作速度を高めるための書込みバッファや、内部マイクロプロセッサの動作の結果とバスＢＵＳのタイミング制約を同期させるタイミング回路などである。またマイクロプロセッサ１０は、システムクロックＳＹＳＣＬＫに基づいてクロック相を生成するクロック発生及び制御回路２０を備える。この例では、クロック発生及び制御回路２０はバスクロックＢＣＬＫとコアクロックＰＣＬＫをシステムクロックＳＹＳＣＬＫから発生させる。

【００２２】図１から明らかなように、マイクロプロセッサ１０は３レベルの内部キャッシュメモリを備える。最高レベルはレベル２キャッシュ１１で、内部バスによりＢＩＵ１２に接続する。この例ではレベル２キャッシュ１１は統一キャッシュで、ＢＩＵ１２を経てバスＢＵＳから全てのキャッシュ化可能なデータとキャッシュ化可能な命令を受け、マイクロプロセッサ１０が与えるバストラフィックの多くはレベル２キャッシュ１１により行われる。またマイクロプロセッサ１０はキャッシュ１１の周りのバストラフィックを制御して、あるバス読取り及び書込みを「キャッシュ化不可」にすることもできる。図１に示すように、レベル２キャッシュ１１は２個のレベル１キャッシュ１６に接続する。レベル１データキャッシュ１６$_d$はデータ専用であり、レベル１命令キャッシュ１６$_1$は命令専用である。マイクロキャッシュ１８は、この例では完全な二重ポートレベル０データキャッシュである。主変換ルックアサイドバッファ（ＴＬＢ）１９は、レベル２キャッシュ１１へのメモリアクセスと、ＢＩＵ１２を経て主メモリへのメモリアクセスを制御する。この制御は、メモリ内のアドレス変換用のページテーブルへのアクセスを整理する。ＴＬＢ１９はページテーブル用のキャッシュでもある。命令マイクロ変換ルックアサイドバッファ（μＴＬＢ）２２とデータマイクロ変換ルックアサイドバッファ（ｕＴＬＢ）３８は、レベル１命令キャッシュ１６$_1$とレベル１データキャッシュ１６$_d$にそれぞれアクセスするために、従来の方法で論理データアドレスを物理アドレスに変換する。

【００２３】図１に示すように、マイクロプロセッサ１０はスーパースカラ型であり、したがって多数の実行ユニットを備える。これらの実行ユニットは、条件付き分岐や整数や論理操作を処理する２個のＡＬＵ４２０及び４２２と、浮動小数点ユニット（ＦＰＵ）３１と、２個のロード・記憶ユニット４００及び４０１と、マイクロシーケンサ４８を備える。２個のロード・記憶ユニット４０はマイクロキャッシュ１８への２個のポートを用いて真の並列アクセスを行い、またレジスタファイル３９内のレジスタへのロード及び記憶操作を行う。この技術で知られているように、レジスタファイル３９はプログラマが用いる汎用レジスタを備え、またコードセグメントレジスタＣＳを含む制御レジスタを備える。

【００２４】これらの多数の実行ユニットは書戻しの際に、それぞれ７段階の多数のパイプラインにより制御される。パイプラインの段階は次の通り。

Ｆ　　取出し：　この段階は命令アドレスを生成して、命令キャッシュ即ちメモリから命令を読み取る。

ＰＤ０　前復号化段階０：　この段階は最大３個の取り出されたｘ８６型の命令の長さと開始位置を決定する。

ＰＤ１　前復号化段階１：　この段階はｘ８６命令バイトを抽出して、復号化のために固定長書式に記録する。

ＤＣ　　復号化：　この段階はｘ８６命令を最小単位動作（ＡＯｐｓ）に変換する。

ＳＣ　　スケジュール：　この段階は該当する実行ユニット（ＦＰＵ３１を含む）に最大４ＡＯｐｓを割り当てる。

ＯＰ　　オペランド：　この段階はＡＯｐｓが示すレジスタオペランドを検索する。

ＥＸ　　実行：　この段階はＡＯｐｓと検索されたオペランドに従って実行ユニットを実行させる。

ＷＢ　　書戻し：　この段階は実行の結果をレジスタまたはメモリに記憶する。

【００２５】図１に戻って、マイクロプロセッサ１０内の種々の機能ブロックは上に述べたパイプライン段階を実行する。取出しユニット２６は命令マイクロ変換ルックアサイドバッファ（μＴＬＢ）２２を用いて、後で説明する分岐予測法などにより命令ポインタから命令アドレスを生成し、レベル１命令キャッシュ１６$_1$に与える。更に後で説明するように、取出しユニット２６は線Ｕ／Ｓを経てコードセグメントレジスタＣＳから、取出しユニット２６での現在の命令のプログラムの種類即ちクラスを示す信号を受ける。更に後で説明するように、取出しユニット２６とロード・記憶ユニット４０の片方または両方との間にパターン履歴バスＰＨＴＢＵＳがあり、この発明の好ましい実施の形態の取出しユニット２６内の１つ以上のパターン履歴表の読取り及び書込みを行うことができる。これについては後で説明する。

【００２６】命令キャッシュ１６$_1$は取出しユニット２６への命令データのストリームを作り、取出しユニット２６は前復号化０段階２８と前復号化１段階３２に、所望のシーケンスで命令コードを与える。これらの２段階は別個のパイプライン段階として動作し、また共に動作して最大３個のｘ８６命令を見つけてデコーダ３４に与える。前復号化０段階２８は３個の可変長ｘ８６命令のサイズと位置を決定し、前復号化１段階３２は多バイト

命令を固定長書式に記録して復号化を容易にする。この例では、復号化ユニット３４は４個の命令デコーダを備え、それぞれは前復号化１段階３２から固定長のｘ８６命令を受けて、１ないし３個の最小単位動作（ＡＯｐｓ）を作る。ＡＯｐｓは実質的にＲＩＳＣ命令と同じである。スケジューラ３６は復号化ユニット３４の出力の復号化待ち行列から最大４つのＡＯｐｓを読んで、これらのＡＯｐｓを該当する実行ユニットに割り当てる。オペランドユニット４４はマルチプレクサ４５を経てスケジューラ３６とマイクロコードＲＯＭ４６から入力を受け、命令の実行に用いるレジスタオペランドを取り出す。更にこの例では、オペランドユニット４４はオペランド転送を行って記憶可能のレジスタに結果を送り、またＡＯｐｓのためにロード及び記憶の種類のアドレスを生成する。

【００２７】マイクロシーケンサ４８とマイクロコードＲＯＭ４６は、ＡＬＵ４２とロード・記憶ユニット４０が、一般に１サイクル内に実行する最後のＡＯｐｓであるマイクロコードエントリＡＯｐｓを実行するのを制御する。この例では、マイクロシーケンサ４８はマイクロコードＲＯＭ４６に記憶されているマイクロ命令を整理して、マイクロコード化されたマイクロ命令に応じて制御を行う。マイクロコード化されたマイクロ命令の例は、複雑なまたは余り用いないｘ８６命令や、セグメントすなわち制御レジスタを修正するｘ８６命令や、例外や割込みの処理や、多サイクル命令（例えばＲＥＰ命令や、全てのレジスタのＰＵＳＨ及びＰＯＰ命令など）などである。

【００２８】またマイクロプロセッサ１０は回路２４を備える。回路２４は、ＪＴＡＧ走査試験の動作やいくつかの内蔵自己試験（ＢＩＳＴ）機能を制御し、製造が完了したときやリセットなどの他の事象のときに、マイクロプロセッサ１０の動作が確実であることを確認する。

【００２９】次に図２を参照して、この発明の好ましい実施の形態の取出しユニット２６の構造と動作を説明する。上に述べたように、取出しユニット２６は復号化のために取り出す次の命令のアドレスを決定する。したがって、取出しユニット２６は命令をマイクロプロセッサ１０のパイプラインに読み込むシーケンスを決定し、この発明のこの実施の形態では、アドレスの推測的な実行、特に分岐予測による実行を制御する。

【００３０】取出しユニット２６の動作は、いくつかの方法の中からマルチプレクサ５２が選択して生成した論理取出しアドレスＦＡに基づいて行う。取出しアドレスＦＡは、復号のために次の順次のアドレスを取り出すときに、単に取出しユニット２６内の取出しポインタ５０の内容から生成する。図２に示すように、取出しポインタ５０は取出しユニット２６内のレジスタであって、その出力はマルチプレクサ５２の１入力と増分器５１に接続する。増分器５１は取出しアドレスの値を進めて次の論理命令の値にし（スーパースカラ機の場合は、次の論理命令は必ずしも次の順次の命令ではない）、進めた取出しアドレスをマルチプレクサ５８の入力に与えて、取出しポインタ５０内に記憶して、またこれを次の取出しに用いる。マルチプレクサ５８は、次のアクセスで用いる取出しポインタ５０の更新された内容のソースを選択する。取出しアドレスＦＡを生成する第２の方法では、例えば取出しユニット２６が予測しなかった行う分岐または予測を誤った分岐の場合に、実行ユニットの１つ（例えばマイクロシーケンサ４８）からマルチプレクサ５２に与える。この値もマルチプレクサ５８の入力に与えて、取出しポインタ５０内に記憶する。

【００３１】また取出しユニット２６は、プログラムシーケンスから次の取出しアドレスＦＡを生成する回路を備える。図２に示すように、取出しユニットは復帰アドレススタック５５を備える。これはいくつかの位置を持つ後入れ先出し（ＬＩＦＯ）メモリで、サブルーチン呼出しとサブルーチン復帰のための復帰アドレスを記憶して、サブルーチンの推測的な実行に用いる。この発明のこの実施の形態では、取出しユニット２６は分岐ターゲットバッファ（ＢＴＢ）５６を更に備える。これはキャッシュと同様なエントリの配置を持ち、分岐命令の現在の段階を予測するのに用いる過去の分岐の履歴を示すデータと、取出しアドレスＦＡとして用いる分岐命令のターゲットアドレスを記憶して、パイプラインをできるだけいつも充てん状態に保つ。この発明のこの実施の形態では、ＢＴＢ５６は２レベル型であり、多数のパターン履歴表（ＰＨＴ）５３と共に動作して、分岐パターン履歴に基づく予測コードを記憶する。これは分岐履歴情報により呼び出す。

【００３２】この発明の好ましい実施の形態に関して後で詳細に述べるように、選択論理８０は、特定のアドレスの分岐予測を生成するのに用いる適当な１つのパターン履歴表５３を、分岐命令を含むプログラムの種類に従って選択する。図２に示すように、選択論理８０は、現在の分岐命令を含むプログラムの種類に関する情報に応じて、パターン履歴表５３の中から選択する。この情報は、例えばコードセグメントレジスタＣＳからの線Ｕ／Ｓにより、また対応する分岐命令のページテーブルエントリのグローバルビットからのグローバルビット線Ｇにより送られる。このようにして、同じ種類のプログラム（例えば、アプリケーションプログラム、共有ライブラリ、オペレーティングシステム機能）により示される分岐行動の類似性を利用することにより、分岐予測の成功率が高くなる。

【００３３】この発明のこの実施の形態では、取出しユニット２６は更にＰＨＴ読取り／書込み回路８３を備える。回路８３は、多数のＰＨＴ５３のそれぞれと、またバスＰＨＴＢＵＳと通信する。後で詳細に説明するように、ＰＨＴ読取り／書込み回路８３はタスク切替えのと

きに、ロード・記憶ユニット４０を経てＰＨＴ５３の選択された１つの内容をメモリとの間で授受する。この発明の好ましい実施の形態では、パターン履歴表はタスク切替えのときに動的に記憶及び書換えが可能なので、特定のタスクの専用のパターン履歴を保持することができる。

【００３４】パターン履歴表５３内の対応する予測コードに基づいて得られる分岐予測に応じて、ＢＴＢ５６はバスＢＲＴＲＧを通してマルチプレクサ５７にターゲット命令アドレスを与える。復帰アドレススタック５５はバスＲＡを通してマルチプレクサ５７に復帰命令アドレスを与える。マルチプレクサ５７の出力はマルチプレクサ５２の第３入力に接続し、またマルチプレクサ５８に接続して取出しカウンタ５０を更新する。このようにマルチプレクサ５２の３入力は次の取出しアドレスＦＡの３つのソースを示す。これらは物理アドレスではなく論理アドレスである。

【００３５】分岐予測の結果は該当する実行ユニットから線ＵＰＤを通して更新論理７０に伝えられる。後で説明するように、更新論理７０はＢＴＢ５６内のエントリの分岐履歴を更新し、また実行して評価された分岐予測が成功か失敗かに応じて、パターン履歴表５３に記憶されている予測コードを更新する。

【００３６】取出しアドレスＦＡは取出しユニット２６内の種々の機能に送られ、復号化のための次の命令の取出しを制御する。例えば、取出しユニット２６は命令μＴＬＢ２２と通信する。命令μＴＬＢ２２は、論理取出しアドレスＦＡと一致する物理アドレスＰＡが前に変換された位置を指す場合は、これを戻す。または論理取出しアドレスＦＡは、取出しユニット２６の外部にある主変換ユニット（図示せず）により物理アドレスに変換される。どちらにしても、取出しユニット２６は命令線アドレスＩＡをレベル１命令キャッシュ１６$_1$に与えて、ここから一連の命令コードを検索する。もちろん、レベル１命令キャッシュ１６$_1$でキャッシュミスが起こった場合は、物理アドレスを統一レベル２キャッシュ１１に与え、またキャッシュミスがこのレベルで起こった場合は、主メモリに与える。命令線アドレスＩＡに応じて、レベル１命令キャッシュ１６$_1$は一連の命令コードシーケンスＣＯＤＥを取出しユニット２６内の命令バッファ及び制御６０に与え、最終的に前復号化０段階２８に与える。この場合は各命令線アドレスＩＡを用いて１６バイトのブロックにアドレスするので、命令バッファ及び制御６０の容量は少なくとも１６バイトである。

【００３７】また取出しユニット２６は、中断命令であると識別された命令を更に取り出さないようにする命令中断チェック回路６２などの他の従来の機能も備える。また取出しユニット２６は、論理取出しアドレスＦＡが現在のコードセグメントの境界の外のアドレスを指すかどうか判断する、コードセグメント限界チェック回路６４を備える。

【００３８】論理取出しアドレスＦＡはＢＴＢ５６の入力に入る。ＢＴＢ５６は、取出しアドレスＦＡが最近取り出した分岐命令を指しているか、また推測的な（ｓｐｅｃｕｌａｔｉｖｅ）実行に用いる分岐履歴をＢＴＢ５６内に記憶しているいるか、を判断する。この技術で知られているように、推測的な実行は、図１のスーパースカラ・マイクロプロセッサ１０などのように深くパイプライン化されたマイクロプロセッサの性能を高める重要な方法である。それは、予測を誤って分岐すると（パイプラインは機能を停止して条件付き分岐の結果を待つ）、実行機会が失われるので大きな損失を生じるからである。ＢＴＢ５６はキャッシュに似た構成に配置されたメモリで、例えば５１２エントリの、４通りのセットアソシエイテイブ・キャッシュバッファである。もちろんＢＴＢ５６は、直接マッピングからフル連想型まで、任意の方法で構成することができる。次に図３を参照して、選択論理８０及び多数のパターン履歴表５３の例と組み合わせて、ＢＴＢ５６の構造を説明する。

【００３９】上に述べたように、この例のＢＴＢ５６は、多数のエントリ６３を有する４通りのセットアソシエイテイブ・キャッシュメモリである。簡単のために、図３では１通りだけを示している。ＢＴＢ５６はセレクタ６１を備える。セレクタ６１は線ＦＡを通して取出しアドレスを受け、取出しアドレスが指すエントリ６３を選択する。セレクタ６１は従来の任意の方法で作り、例えばデコーダやタグ比較器や簡単なマルチプレクサにより、取出しアドレスからＢＴＢ５６内のエントリ６３を選択する。ＢＴＢ５６内の各エントリ６３は特定の分岐命令の論理取出しアドレスＦＡにより各エントリ６３を識別するのに用いるタグフィールドＴＡＧを持ち、セレクタ６１はこれと入力する論理取出しアドレスＦＡの一部とを比較する。この技術で知られているように、タグフィールドＴＡＧは対応する分岐命令の論理取出しアドレスＦＡの選択されたビットを直接記憶し、またはこれらの選択された論理アドレスビットの論理的組合わせに対応する。一般にタグフィールドＴＡＧは線アドレスと、取出し線内の命令のバイトオフセットを示すオフセットを含む。またＢＴＢ５６内の各エントリ６３は、分岐命令ターゲットアドレスの論理アドレスを含むターゲットフィールドＴＡＲＧＥＴを有する。上述のように、「行う」と予測された分岐命令と一致する、エントリ６３のＴＡＲＧＥＴ部内のターゲットアドレスは、ＢＴＢ入出力論理６９からバスＢＲ　ＴＲＧを通してマルチプレクサ５７に送られる。分岐が「行わない」である場合は、マルチプレクサ５２は、単に次の順次の論理アドレス（即ち、取出しポインタ５０の内容）を次の論理取出しアドレスＦＡとして選択する。

【００４０】またこの発明のこの実施の形態の各エントリ６３は、タグフィールドＴＡＧに対応する分岐命令の

分岐履歴を記憶するｍビットの分岐履歴フィールドＢＨを含む。分岐履歴フィールドＢＨに記憶される分岐履歴は、命令の実行が終わったときに決定される関連する分岐命令の実際の分岐履歴と、まだ実行が終わっていない分岐命令の事例の予測結果から成る推測的な分岐履歴を含む。更に、同時係属出願の米国仮出願番号第６０／０２０，８４４号、１９９６年６月２８日出願（ここに参照することにより挿入する）に述べられているように、ＢＴＢ５６内の各エントリ６３は分岐履歴フィールドＢＨ内の推測的な分岐履歴ビットの数を示すカウンタも備え、予測誤りから回復するのに用いる。またＢＴＢ５６内の各エントリ６３は標識ＴＹＰＥを含む。これはその関連する命令の分岐命令の種類（即ち、条件付き分岐、ＣＡＬＬ、ＪＵＭＰ、ＲＥＴＵＲＮ）を記述するもので、分岐を予測するのに用いる。ＣＡＬＬ、ＪＵＭＰ、ＲＥＴＵＲＮなどの無条件分岐は常に「行う」と予測する。ＬＲＵビットや有効ビットやその他の制御ビット（図示せず）などの追加のビットもＢＴＢ５６の各エントリ６３内に与えられる。

【００４１】図２に関して上に述べたように、多数のパターン履歴表（ＰＨＴ）５３を用いて、選択されたＢＴＢのエントリ６３の分岐履歴フィールドＢＨの最新のｋビットに基づいて条件付き分岐の行動を予測する。この発明のこの実施の形態では、各ＰＨＴ５３は特定の種類のプログラムに関連し、分岐履歴フィールドＢＨはＰＨＴ５３の任意の１つにアクセスすることができる。しかし予測コードは、命令を取り出した種類のプログラムに該当する１つのＰＨＴ５３だけから選択する。図２ではＰＨＴ５３とＢＴＢ５６は物理的に別の回路で実現しているが、もちろん必要に応じてＰＨＴ５３をＢＴＢ５６に含めてよい。図３で明らかなように、この例では４個のＰＨＴ５３$_3$から５３$_0$をＢＴＢ５６と組み合わせて実現している。

【００４２】各ＰＨＴ５３は簡単なルックアップメモリであって、それぞれはＢＴＢ５６の選択されたエントリ６３からｋビットの分岐履歴を受けるセレクタ６７を備え、これに対応する２^{k}個の予測エントリＰＲＤの中の１つを選択する。セレクタ６７はデコーダまたはマルチプレクサで実現してこの機能を実行する。図３に示すように、ＰＨＴ５３$_3$から５３$_0$はそれぞれ選択されたエントリ６３からｋビットの分岐履歴を受け、与えられたｋビットの分岐履歴に対応するエントリＰＲＤの内容に対応する一組の線ＰＲＥ$_3$からＰＲＥ$_0$を通してパターン履歴コードを出す。選択された１つのＰＨＴ５３に指標を付ける際に、例えばいくつかのアドレスヒットや制御情報などの他の情報を、分岐履歴フィールドＢＨのこれらのｋビットと組み合わせてよい。この発明のこの実施の形態では、各組のＰＲＥは２線を用いて、４状態の分岐予測モデル（即ち、強く行う、行う、行わない、強く行わない）の下に、従来の方法で２ビットのパターン履歴コードを送る。

【００４３】この発明のこの実施の形態では、選択論理８０はＢＴＢ５６に与えられる選択された１つのＰＨＴ５３の出力を送る回路を備える。もちろん選択論理８０は多数のＰＨＴ５３の中の適当な１つに選択的にアドレスするようにして実現してよい。この例では、ＰＨＴ５３$_3$乃至５３$_0$からパターン履歴線ＰＲＥ$_3$乃至ＰＲＥ$_0$をそれぞれマルチプレクサ６８の入力に与える。マルチプレクサ６８はパターン履歴線ＰＲＥ$_3$乃至ＰＲＥ$_0$の組の１つを選択して、線ＴＮＴを通してＢＴＢ入出力論理６９に与え、これから適当な分岐予測を行う。この発明のこの実施の形態では、マルチプレクサ６８は、現在の分岐命令に対応するページテーブルエントリＰＴＥ$_j$（後で詳細に説明する）内のグローバルビットＧの状態に応じて、また線Ｕ／Ｓ上の信号に従って制御される。この例では、線Ｕ／Ｓ上の信号は、マイクロプロセッサ１０のコードセグメントＣＳレジスタに含まれる現在の特権レベル（ＣＰＬ）の状態に対応する。この例では、マイクロプロセッサ１０はｘ８６アーキテクチャに従って作る。後の説明から明らかなように、マルチプレクサ６８は分岐予測を行うのに用いる線ＰＲＥ$_3$乃至ＰＲＥ$_0$の適当な組を、分岐命令を出すプログラムの種類即ちクラスに従って選択する。これについては以下に説明する。

【００４４】ｘ８６アーキテクチャでは、マイクロプロセッサ１０が実行するプログラムはカーネル（最高の特権）からアプリケーション（最低の特権）まで、異なる特権レベルに従って分類される。したがって、個々の命令は、種々の特権レベルに従ってアクセスが保護されるメモリ部分に常駐する。これにより、マルチタスク環境で動作する多数のアプリケーションプログラムはプログラムやサブルーチンを共有することができる。ｘ８６アーキテクチャのメモリページング保護機構では、メモリのこれらの部分をユーザ及びスーパーバイザレベルと呼ぶ。ユーザ保護レベル（ＣＰＬ＝３）はアプリケーションプログラムが記憶されているメモリ位置に割り当て、スーパーバイザ保護レベル（ＣＰＬ＝０からＺ）はオペレーティングシステムや拡張やドライバやカーネルが常駐するメモリ位置に割り当てる。したがってこの例では、線Ｕ／Ｓ上の信号は、コードセグメントＣＳ内のＣＰＬの値に基づいて、現在の分岐命令を含むプログラムの特権レベルを示す。

【００４５】もちろん現在の分岐命令が関連するプログラムの種類は、他の方法で、例えばｘ８６アーキテクチャ内のＣＰＬに対応する多数の信号線により、または他のアーキテクチャに従うマイクロプロセッサ内の他の種類の信号により、示してよい。どちらにしても、マルチプレクサ６８は現在のプログラムの種類に対応する少なくとも１つの信号に従って制御され、分岐行動は異なる種類のプログラムの分岐命令によって異なってよい。こ

の発明の好ましい実施の形態では、ＢＴＢ５６とパターン履歴表５３は、アプリケーションプログラム内の分岐命令（ユーザレベルのメモリに常駐している命令）の分岐行動の類似性や、オペレーティングシステム内の分岐命令（スーパーバイザレベルのメモリに常駐している命令）の分岐行動の類似性や、両方のレベルの共有ルーチン内の命令の分岐行動の類似性や、これらの異なる種類のプログラム内の分岐命令間の分岐行動の非類似性などを利用する。これを実現するために、この発明の好ましい実施の形態では、少なくとも１つのＰＨＴ５３をユーザレベルの分岐命令に関して用いるように、また少なくとも１つの他のＰＨＴ５３をスーパーバイザーレベルの分岐命令に関して用いるように割り当てる。この実施の形態では、２つのＰＨＴ５３をユーザレベルの分岐に割り当て、他の２つのＰＨＴをスーパーバイザレベルの分岐命令に割り当てる。この発明のこの実施の形態では、線Ｕ／Ｓ上の信号をマルチプレクサ６８の制御入力の１つに与えてこの制御を行う。

【００４６】上に述べたように、線Ｕ／Ｓの状態と他の制御フラグ及びビットを組み合わせ、これを用いて適当なＰＨＴ５３を選択してよい。マイクロプロセッサの分野でよく知られているように、また上に述べたように、ページテーブルエントリを用いて論理アドレスから物理アドレスへのアドレス変換を行う。上に述べたようにマイクロプロセッサ１０では、ＴＬＢ１９はページテーブルエントリＰＴＥのキャッシュとして動作する。それぞれは、現在のアドレスが写像するページフレームアドレスを含むだけでなく、アドレスが指すメモリのページフレームに関するこの技術で知られているいくつかの制御情報も含む。ペンティアム（Ｒ）プロファミリー・デベロッパーズマニュアル（Ｐｒｏ　Ｆａｍｉｌｙ　Ｄｅｖｅｌｏｐｅｒ’ｓ　Ｍａｎｕａｌ）、第３巻、オペレーティングシステム・ライターズガイド（Ｏｐｅｒａｔｉｎｇ　Ｓｙｓｔｅｍ　Ｗｒｉｔｅｒ’ｓ　Ｇｕｉｄｅ）（インテル、１９９６年）の３－２１ページから３－２６ページ（参照することによりここに挿入する）に述べられているように、ＰＥＮＴＩＵＭ　ＰＲＯマイクロプロセッサのアーキテクチャに従うページテーブルエントリはグローバル（ページ）ビットＧを含む。このビットが、セットされているときは、変換ルックアサイドバッファ内のページエントリはタスク切替えのときにクリアされないことを示す。これにより、いくつかのタスクからのアクセスが可能な、メモリの共通ページを割り当てることができる。

【００４７】例えば、Ｃ＋＋言語で書かれたプログラム用のライブラリルーチンは、多数のＣ＋＋タスクからアクセスできるようにグローバルメモリページ内に記憶する。Ｃａｌｄｅｒ　と　Ｇｒｕｎｗａｌｄの論文に関して上に述べたように、ライブラリルーチンは他の種類のプログラムとは異なる分岐行動を持つことが分かった。したがって、適当なＰＨＴ５３の選択にグローバルビットを用いれば、ページテーブルエントリにグローバルビットを持つマイクロプロセッサに有用である。この発明のこの実施の形態では、ページテーブルエントリＰＴＥｊ内のグローバルビットＧの状態（上に述べたように、タスク切替えのときにそのページエントリがＴＬＢ１９からクリアされるかどうかを示す）は、上に述べたコードセグメントＣＳから線Ｕ／Ｓで送られて来る信号と共に、マルチプレクサ６８の制御入力に入る。

【００４８】この発明の好ましい実施の形態の、マルチプレクサ６８によるＰＨＴ５３$_3$から５３$_0$の選択の例を次の真理値表に示す。

【表１】

ユーザ／スーパーバイザ User/Supervisor	グローバルビットGの状態 Global bit G state	PHT selected 選択されたPHT
Supervisor スーパーバイザ	0	53_0
Supervisor スーパーバイザ	1	53_1
User ユーザ	0	53_2
User ユーザ	1	53_3

【００４９】この発明の好ましい実施の形態では、マルチプレクサ６８の制御は、線Ｕ／Ｓ上のユーザ／スーパーバイザ状態と、グローバルビットＧの状態に応じて行うが、適当なＰＨＴ５３を選択するのに他の制御信号または情報をこれらの代わりにまたは追加して用いてよい。例えば、線ＦＡ上の取出しアドレスの一部と書込み可能な範囲レジスタの内容とを比較して、取出しアドレスが、範囲レジスタが示す範囲内か範囲外かを決定し、適当なＰＨＴ５３を選択する際のプログラムの種類の識別子としてこれを用いてよい。または、ページテーブルエントリＰＴＥ内のこれまで割り当てられていない他のビットを用いて、適当なＰＨＴ５３の選択のプログラム制御を行うことができる。更にまた、セグメント記述子ＤＥＳＣ内の１つ以上のビットの状態に従ってこの選択を行ってもよい。セグメント記述子ＤＥＳＣは、ｘ８６アーキテクチャのマイクロプロセッサの保護モード動作中にセグメントセレクタが指標付けした、グローバルまたはローカル記述子テーブル内のエントリである。これ

らのビットは、現在まだ定義されていないビットか、またはセグメント記述子の拡張から得られたＰＨＴ５３の選択コードを与えるビットである。更にまた、上述の制御信号と取出しアドレス自身の選択されたサブセットとを組み合わせて、分岐予測に用いる適当なＰＨＴ５３を選択してよい。多数のＰＨＴ５３から選択するこれらの方法は、出願人の仮出願番号第　／号、１９９６年１２月１０日出願、「マイクロプロセッサ内で分岐予測に用いる多数のグローバルパターン履歴表（Ｍｕｌｔｉｐｌｅ　Ｇｌｏｂａｌ　Ｐａｔｔｅｒｎ　Ｈｉｓｔｏｒｙ　Ｔａｂｌｅｓ　ｆｏｒ　Ｂｒａｎｃｈ　Ｐｒｅｄｉｃｔｉｏｎ　ｉｎ　ａ　Ｍｉｃｒｏｐｒｏｃｅｓｓｏｒ）」（代理人書類番号第ＴＩ－２３７９１Ｐ）に詳細に述べられており、これを参照することによりここに挿入する。

【００５０】図３に戻って、前に述べたように、線ＴＮＴ上のマルチプレクサ６８の出力はＢＴＢ入出力論理６９に与えられる。線ＴＮＴ上の予測コードが「予測された行う」分岐を示す場合は、ＢＴＢ入出力論理６９はＢＴＢ内の現在のエントリ６３のＴＡＲＧＥＴ部に対応する有効な分岐ターゲットアドレスを与える。またＢＴＢ入出力論理６９は命令バッファ及び制御６０への線ＡＴＲに、現在の命令の対応する分岐予測を出す。更にＢＴＢ入出力論理６９は、実行ユニットからの線ＮＥＷＮにより、新しく出現した分岐命令の該当するタグ、ターゲット、オフセット、種類、履歴情報を受け、選択されたエントリ６３にこの情報を従来の方法で書き込む。この技術で知られているように、ＬＲＵなどの制御ビットを用いて、新しい命令の情報を書き込むエントリ６３を選択する。

【００５１】またＢＴＢ５６は更新論理７０を備える。更新論理７０は、前に予測された分岐命令の結果を示す実行ユニット（例えば、ＡＬＵ４２）からの信号をバスＵＰＤに受ける。更新論理７０は従来の回路で、関連する分岐の予測が正しかったか誤りだったかに従って、ＢＴＢ５６内のエントリ６３の内容を更新する。更に、ＰＨＴ５３が適応性を持つことを考慮して、更新論理７０はＰＨＴ５３への線ＰＨＵを駆動して、従来の方法により、実行された分岐命令の予測の結果に従って予測コードエントリＰＲＤの内容を更新する。しかしこの発明の好ましい実施の形態では、更新論理７０が生成した線ＰＨＵ上の信号は、多数のＰＨＴ５３の中から、終わった分岐について更新すべき適当なものを選択する。パターン履歴表の適応更新はこの技術で知られており、例えば前に参照したＹｅｈ　とｐａｔｔの論文に述べられている。

【００５２】この発明の好ましい実施の形態のＢＴＢ５６と多数のグローバルパターン履歴表５３の動作を、図３に関して以下に説明する。もちろん、ＢＴＢ５６は線ＦＡのアドレスで取り出された非分岐命令では動作しない。最近出現しなかった（そしてＢＴＢ５６内に割り当てられた有効なエントリ６３をこの時点に持たない）分岐命令については、セレクタ６１はエントリ６３のどのＴＡＧフィールドにも一致するタグを見出さないので、誤り信号、即ち「失敗」信号を命令バッファ及び制御６０への線ＡＴＲに返す。この場合は、有効な分岐ターゲットアドレスはマルチプレクサ５７へのバスＢＲ　ＴＲＧに与えられず、マルチプレクサ５２は次の論理取出しアドレスＦＡのために別のソース（一般に取出しポインタ５０）を選択する。この分岐命令の実行段階が終わると、ＢＴＢ５６は線ＮＥＷＮから得た情報を用いて、ＢＴＢ入出力論理６９を経て従来の方法で更新され、有効なエントリ６３がこの分岐命令に割り当てられる。

【００５３】前に出現した、したがって対応するエントリ６３をＢＴＢ５６内に持つ（即ち、取出しアドレスＦＡの一部がエントリ６３のＴＡＧフィールドと一致し、対応するエントリ６３のＴＹＰＥ部から分かる）無条件分岐命令では、ＢＴＢ５６は命令バッファ及び制御６０への線ＡＴＲに「行う」予測を与え、またマルチプレクサ５７へのバスＢＲ　ＴＲＧにこのエントリ６３のＴＡＲＧＥＴフィールドからターゲットアドレスを与え、マルチプレクサ５２はこれを次の命令アドレスのソースとして、従来の方法で用いる。これも無条件分岐命令であるサブルーチンＲＥＴＵＲＮ命令の場合は、マルチプレクサ５７は復帰アドレススタック５５からの線ＲＡ上の適当な復帰アドレスを選択し、マルチプレクサ５２に次の命令アドレスのソースとして、従来の方法で与える。

【００５４】ＢＴＢ５６のセレクタ６１が、線ＦＡで送られる現在の取出しアドレスが有効なエントリ６３を持つ条件付き分岐命令に対応すると決定した場合は、ＢＴＢ５６は有効なエントリ６３のｋビットの分岐履歴フィールドＢＨを各ＰＨＴ５３$_3$から５３$_0$に送る。これらのｋビットはその分岐命令の最近のｋ個の予測に対応し、実際の分岐結果だけを含むこともあり、またはまだ評価が済んでいない推測的な分岐予測も含むこともある。選択されたエントリ６３のこれらのｋビットの分岐履歴フィールドＢＨを、一般に現在の分岐命令の現在の分岐パターンと呼ぶ。この発明の好ましい実施の形態では、各ＰＨＴ５３$_3$から５３$_0$内のセレクタ６７はこれらのｋビットを復号して、現在の分岐パターンと一致する予測コードエントリＰＲＤを選択し、選択された予測コードエントリＰＲＤの内容を、関連する出力線ＰＲＥ$_3$からＰＲＥ$_0$によりマルチプレクサ６８に送る。各予測コードエントリＰＲＤは好ましくは２ビットコードを含み、行う、行わない、強く行う、強く行わない、の４つの予測状態の中の１つを示す。

【００５５】一方、分岐命令を含むメモリの対応する部分に関するページテーブルエントリＰＴＥ$_j$内のグローバルビットＧの状態と、線Ｕ／Ｓの状態は、マルチプレクサ６８を制御して一組の出力線ＰＲＥ$_3$からＰＲＥ$_0$

を選択し、線ＴＮＴを経てＢＴＢ５６のＢＴＢ入出力論理６９に与える。上に述べたように、線ＴＮＴは好ましくは、行う、行わない、強く行う、強く行わない、の予測状態の１つを示す２ビットコードを送る。次にＢＴＢ入出力論理６９は線ＴＮＴ上のコードに基づいて予測を得、この予測（「行う」または「行わない」）を線ＡＴＲで命令バッファ及び制御６０に送る。予測が「行う」の場合は、対応するエントリ６３のＴＡＲＧＥＴフィールドをバスＢＲＴＲＧに与え、マルチプレクサ５７と５２はこれを次の論理取出しアドレスＦＡとして選択する。予測が「行わない」の場合は、バスＢＲ　ＴＲＧに有効なターゲットアドレスを与えず、マルチプレクサ５２は取出しポインタ５０の増分された出力を次に取り出す命令のアドレスとして選択する。予測を生成した後で、かつＢＴＢ５６がエントリ６３の分岐履歴フィールドＢＨ内に推測的な分岐履歴を記憶している場合は、更新論理７０は現在の命令に対応するエントリ６３内の分岐履歴フィールドＢＨを更新する。命令に関する識別情報と、予測の生成に用いるＢＴＢ５６内のエントリ及びＰＨＴ５３に関する識別情報と共に、現在の分岐命令に関する予測情報も、パイプラインにより命令と共に送られる。または現在の分岐命令の小さな識別子をパイプラインと共に送ってよい。この場合はこの識別子は、ＢＴＢ５６と該当するＰＨＴ５３を更新するのに用いる取出しユニット２６のまたはその近くのローカル記憶内の位置を指す。

【００５６】分岐命令が終わると、該当する実行ユニットは分岐の実際の結果を線ＵＰＤを通して更新論理７０に送る。更新論理７０は終わった命令に対応するＢＴＢ５６のエントリ６３内の分岐履歴フィールドＢＨに向けて適当な信号を生成して、対応する予測が正しいか正しくないか検査する。更に更新論理７０は分岐の実際の結果に従って、線ＰＨＵにより、該当するＰＨＴ５３の中の該当する予測コードエントリＰＲＤを更新する（必然的に線ＰＨＵには正しいＰＨＴ５３と該当するエントリＰＲＤを選択するのに必要な信号が送られる）。

【００５７】図３に示すようにこの発明の好ましい実施の形態では、ＰＨＴ読取り／書込み回路８３の制御の下に、ＰＨＴ５３はバスＰＨＴＢＵＳを経て読取り可能かつ書込み可能である。この発明のこの実施の形態では、ＰＨＴ読取り／書込み回路８３は双方向マルチプレクサ８２とＰＨＴアクセス制御回路８４を備える。マルチプレクサ８２の片側はバスＰＨＴＢＵＳに接続し、その反対側は個々のバスによりＰＨＴ５３に接続する。この例では、３２ビットバスによりマルチプレクサ８２とＰＨＴ５３を接続し、１つのＰＨＴ５３の全内容を１動作で送ることができる。またはより小さなバス（２ビットバスＰＲＥを含む）を用いて、複数の読取りまたは書込みサイクルにより、選択されたＰＨＴ５３の内容をマルチプレクサ８２に送ってよい。この場合は、好ましくはマルチプレクサ８２と共に別のレジスタを設けて、選択されたＰＨＴ５３の全内容をバスＰＨＴＢＵＳのデータ線に乗せる。

【００５８】この例では、マルチプレクサ８２は３２ビットバスにより各ＰＨＴ５３$_1$からＰＨＴ５３$_3$に接続する。上に述べたように、ＰＨＴ５３$_0$はスーパーバイザレベルのプログラムに関連するがグローバルではない。したがってこの発明のこの実施の形態では、ＰＨＴ５３$_0$は好ましくは書込み可能でない。それは、この性質のプログラムの分岐パターン履歴に基づく予測コードは、ＢＴＢ５６内に保持するのが好ましいからである。もちろん別の方法として、図３に点線のバス線で示すように、ＰＨＴ５３$_0$をマルチプレクサ８２に同様に接続してよい。各ＰＨＴ５３$_1$からＰＨＴ５３$_3$は書込み可能であって、図３に示すようにマルチプレクサ８２に接続する。ＰＨＴ５３$_2$と５３$_3$は、関連するユーザ特権レベルを与えられた種々のタスクで動的に書き換えるのに特に適している。更に、ＰＨＴ５３$_1$は、関連するグローバルスーパーバイザレベルのプログラム（一般にライブラリ）が共通の分岐行動を持つという特殊な場合だけ書換えてよい。特に、ＰＨＴ５３$_1$の内容を保存したり再ロードしたりするという特殊な場合は、ライブラリルーチンの種類がタスクによって変わる場合に対応する（例えば、Ｃ＋＋タスクとＣＯＢＯＬタスクの間で切り替える）。

【００５９】ＰＨＴアクセス制御回路８４はマルチプレクサ８２を制御して、バスＰＨＴＢＵＳのデータ線と読み取りまたは書き込む選択されたＰＨＴ５３を接続し、また関連する読取り／書込み線Ｒ／Ｗを経て選択されたＴＰＨＴ５３を制御して、読取りまたは書込みを行わせる。ＰＨＴ５３の選択と、読取りと書込みのどちらを行うかの選択は、バスＰＨＴＢＵＳの制御線によりＰＨＴアクセス制御回路８４に伝えられる。この発明の好ましい実施の形態では、ＰＨＴアクセス制御回路８４は選択された読取り／書込み線Ｒ／Ｗに該当する信号を出し、選択されたＰＨＴ５３は読取りか書込みかに従って、その全内容をマルチプレクサ８２に送り、またはマルチプレクサ８２から新しい内容を受ける。またＰＨＴアクセス制御回路８４はオペランドユニット４４に適当なハンドシェーク信号を送り、バスＰＨＴＢＵ上のデータの送信を制御する。

【００６０】この発明の好ましい実施の形態では、１つ以上のＰＨＴ５３の内容の読取りと書込みは、好ましくはタスク切替えのときに行う。マルチタスクのマイクロプロセッサの技術で基本的なことであるが、タスク切替えは、現在の活動的なタスクを他のタスクにより中断する事象である。中断されたタスクはタスク切替えのときにその全ての条件を、一般にこの技術でタスク制御構造と呼ぶ（またはタスク制御ブロックとかタスク状態ブロックとも呼ぶ）メモリの一部に保存する。特定すると、

上述のようにマイクロプロセッサ１０を実現するｘ８６アーキテクチャでは、タスク制御構造は、中断されたタスクの条件を記憶するタスク状態セグメント（ＴＳＳ）を含む。タスク状態セグメントの特定の位置は変動してよく、一般に主メモリ３０５内に常駐し、実行中の便宜のためにその写しをレベル２キャッシュ１１やその他のキャッシュに記憶する。後でタスク切替えを行って前に中断されたタスクを再開するときなどに、そのタスクのＴＳＳの内容を検索してマイクロプロセッサの適当なレジスタや記憶に読み込み、タスクを再開する。この技術でよく知られているように、任意の時点で活動的なタスクは１つだけであるが、このようなマルチタスク動作を行うとマルチ処理のように見える。

【００６１】この発明の好ましい実施の形態では、タスク切替えのときに、１つ以上のＰＨＴ５３の内容をＴＳＳの一部に記憶し、タスクを再び活動的にするタスク切替えのときにそこから検索して、対応するＰＨＴ５３に再び読み込む。このようにして、この発明の好ましい実施の形態のマイクロプロセッサ１０は、分岐パターン履歴に基づく分岐予測情報をタスク毎に保持するので、より正確に分岐予測を行うことができる。

【００６２】図４を参照して、マイクロプロセッサ１０が実行する特定のタスクに関連する、この発明の好ましい実施の形態の例示のＴＳＳ９０のメモリマップを詳細に説明する。もちろんマイクロプロセッサ１０が実行する各タスクは自身のＴＳＳ９０をメモリ内に持ち、各ＴＳＳ９０は、図４に示すように、またこの発明の好ましい実施の形態に従って説明するように配列される。ＴＳＳ９０内の各語はＴＳＳベースアドレスから或るオフセットにある。この例では、ＴＳＳ９０はメモリの中の３０語を占め、オフセットはＴＳＳ９０のベースアドレスから最大２９語である。ＴＳＳ９０の内容の多くは、特に２５語以下のオフセットにあるＴＳＳ９０の内容は、インテル社製のＰＥＮＴＩＵＭマイクロプロセッサの機能性を持つマイクロプロセッサでは従来からあるものである。図４に示すように、ＴＳＳ９０は完全なオフセット及びセグメントレジスタの内容と、異なる特権レベル（ＣＰＬ０からＣＰＬ２）のスタックのＥＳＰポインタ及びセグメントＳＳと、タスクのページディレクトリのベースアドレスを記憶するＣＲ３レジスタの保存された内容を含む。またＴＳＳ９０は、Ｉ／Ｏマップベースエントリを含む。これは、保護モードにおいてＩ／Ｏアドレス空間の保護に用いるＩ／Ｏマップのアドレスである。前のＴＳＳに逆に連結するためのエントリはセグメント記述子を含む。これは、タスクが互いに入れ子になっているときに前の中断されたタスクのＴＳＳを参照するものである。Ｔビットはデバッグトラップビットで、これがセットされると、タスク切替えの時にデバッグ例外になる。ＴＳＳ９０の中のＮ／Ｕと示されているフィールドは用いない。

【００６３】上に述べた従来のＴＳＳエントリの他に、ＴＳＳ９０はこの発明の好ましい実施の形態の、分岐パターン履歴に基づく予測情報の記憶と検索に用いる別のエントリを含む。これらの別のエントリはＴＳＳベースアドレスから２６語オフセットから始まる。ＴＳＳベースアドレスから２６語オフセットの下位部分は、動的ロード制御ビットＤＬＢを有するエントリ９２を含む。ＴＳＳのエントリ９２内のＤＬＢのビット数は書込み可能なＰＨＴ５３の数に対応し、ＤＬＢの各ビットは１つのＰＨＴ５３に関連する。ＰＨＴ５３$_1$から５３$_3$だけが書込み可能（ＰＨＴ５３$_0$は書込み不可）である図３の例では、エントリ９２内に３ビットのＤＬＢが与えられる。後で詳細に説明するように、ＤＬＢの各ビットは、ＴＳＳ９０に関連するタスクにタスク切替えを行うときにその関連するＰＨＴ５３に専用のパターン履歴データをロードするかどうかを示す。一般にＤＬＢのビットは、必要に応じてタスク自身による、またはオペレーティングシステムによる命令制御の下にセットされる。

【００６４】またＴＳＳ９０は、この例でＴＳＳ９０内の２７語から２９語オフセットにあるエントリ９１$_1$から９１$_3$を含む。これらは書換え可能なＰＨＴ５３$_3$から５３$_1$にそれぞれ対応する。この例でエントリ９１は３２ビットのサイズを持ち、それぞれ関連するＰＨＴ５３の内容を記憶する。その内容は、ＴＳＳ９０に関連するタスクを中断した最近のタスク切替えのときの条件である。上に述べたように、この場合は分岐履歴の４ビットをＰＨＴ５３に指標付けし、また各ＰＨＴ５３はそれぞれ各２ビットの１６エントリを含んでいるので、１つのＰＨＴ５３の全内容を記憶するには３２ビットの記憶で十分である。

【００６５】２６語オフセットにある語の高位部分のエントリ９４は、ＰＨＴ５３の内容を含むＴＳＳ９０の部分の、ベースアドレスに対応するフィールドを含む。したがって、エントリ９１のベースアドレスはＴＳＳ９０のベースアドレスとエントリ９４の内容の和のところである。図４に示す例では、エントリ９４の内容は２７語オフセットに対応する。または、エントリ９１の位置はＴＳＳ９０の中の別の場所でよい。この場合は、エントリ９４の内容は、ＴＳＳ９０のベースアドレスに対するエントリ９１の位置を示す。

【００６６】次に図５を参照して、この発明の好ましい実施の形態のタスク切替えルーチンの一部の動作を説明する。図５の動作は、タスク切替えに必要な適当なシステムレベルの動作、例えば中断されたタスクに関するＴＳＳ内の機械条件の情報の記憶や、新たに活動化するタスクに必要なＴＳＳからの機械条件の再ロードなど、を行う一連のプログラム命令内に含まれる。したがって、図５の動作はマイクロプロセッサ１０内の制御及び実行回路により行われる。これは従来のオペレーティングシステムの命令シーケンスで一般的である。以下の説明の

便宜上、中断されたタスクをＴＳＳ９０ｉに関連したタスクｉと呼び、新たに活動化するタスクをＴＳＳ９０ｊに関連したタスクｊと呼ぶ。

【００６７】図５に示すタスク切替えプロセスの部分は決定９５から始まる。ここで、中断されるタスクｉのＴＳＳ９０ｉ内のビットＤＬＢの状態を調べる。上に述べたように、ＴＳＳ９０ｉ内のＬＢＤのビットがセットされているときは、ＤＬＢのセットされたビットで示されるＰＨＴ５３については、タスクｉは活動的なときの分岐予測に自身の分岐パターン履歴に基づく予測情報を用いることを示す。したがって、専用の予測情報を他のタスクにより修正されるのを防ぐために、示されたＰＨＴ５３の現在の条件をＴＳＳ９０ｉに記憶する。したがって、ＴＳＳ９０ｉ内のＤＬＢの任意のビットがセットされている（即ち、決定９５がＹＥＳである）場合は、プロセス９６を行う。図３を参照すると、プロセス９６でＰＨＴアクセス制御８４はマルチプレクサ８２を制御して、ＴＳＳ９０ｉ内のＤＬＢのセットされているビットにより示されたＰＨＴ５３を順次にバスＰＨＴＢＵＳに接続し、また選択されたＰＨＴ５３に関連するセレクタ６７を制御して、その関連するＰＨＴ５３の内容を順次に読み取る。これらの内容を順次にバスＰＨＴＢＵＳに乗せ、メモリ内の、中断されたタスクｉに関連するＴＳＳ９０ｉの対応するエントリ９１に記憶する（ロード・記憶ユニット４０の１つにより）。上に述べたように、この情報を記憶するＴＳＳ９０ｉ内のアドレスは、ＴＳＳベースアドレスと、エントリ９４のＰＨＴエリアベースの和から決定する。次に流れは決定９７に進む。ＴＳＳ９０ｉ内のＤＬＢのどのビットもセットされていない（即ち、決定９５がＮＯである）場合も、流れは決定９７に進む。

【００６８】決定９７では、開始（または再開）するタスクｊのＴＳＳ９０ｊ内のＤＬＢのビットの状態を決定する。ＴＳＳ９０ｊ内のＤＬＢのどのビットもセットされていない（決定９７がＮＯである）という条件は、タスクｊがＰＨＴ５３の内容を現在の条件で用いることを示す。したがって、どのＰＨＴ５３も操作せずに流れはタスク切替えルーチンに進む。ＴＳＳ９０ｊ内のＤＬＢの１つ以上のビットがセットされている（決定９７がＹＥＳである）場合は、タスクｊは自身の分岐パターン履歴に基づく予測情報を持ち、これをその分岐命令の分岐予測に用いる。この場合は、流れはプロセス９８に進む。

【００６９】プロセス９８で、ロード・記憶ユニット４０の１つがタスクｊのＴＳＳ９０ｊの対応するエントリ９１の内容をバスＰＨＴＢＵＳに乗せると同時に、ＰＨＴアクセス制御回路８４は、決定９７でＤＬＢのビットがセットされていると決定したＰＨＴ５３のセレクタ６７に書込み信号を順次出す。プロセス９８で、ＰＨＴアクセス制御回路８４とマルチプレクサ８２は、決定９７で示されたＰＨＴ５３のそれぞれについて書込み動作を繰り返す。各繰返しにおいて線Ｒ／Ｗを通して書込み信号をセレクタ６７に与えると、選択されたＰＨＴ５３にＴＳＳ９０ｊの対応するエントリ９１の内容がロードされ、タスクｊを中断したタスク切替えのときに前に記憶した分岐パターン履歴に基づく予測コードを持つ。多数のＰＨＴ５３をこのようにして書き換えるので、同じタスク内の異なるプログラムの種類（例えば、アプリケーションコードやライブラリ）の分岐命令は、タスク特有の分岐パターン履歴に基づく予測情報に基づいて予測することができる。プロセス９８の後、流れは適当なタスク切替えルーチンに戻り、従来の方法でタスク切替えプロセスを終わる。

【００７０】したがってこの発明の好ましい実施の形態の動作の結果、分岐パターン履歴に基づく予測情報は各タスクの専用の形式で保持され、他のタスク内の命令の分岐行動により予測情報が修正されるのを防ぐ。更にこの発明の好ましい実施の形態では、分岐パターン履歴に基づく予測情報の記憶と検索はタスク切替えのときに自動的に行われるので、プログラマがプログラム制御により行わせる必要はない。このようにこの発明により分岐予測性能が改善される。

【００７１】この発明については種々の別の形態が考えられる。例えば、多数のＰＨＴを用いることが好ましいが、この発明は、極端な場合は単一のグローバルＰＨＴを用い、また逆の極端な場合は各ＢＴＢエントリ毎に１つのＰＨＴを用いるなど、他のＢＴＢ構成で用いることができる。しかし単一のＰＨＴを用いる場合は、タスクの開始のときに分岐パターン履歴に基づく予測情報を与える必要がある。このため、例えば新しいタスクへのタスク切替えのときにＰＨＴの内容を記憶する。しかしそのタスクが分岐を予測するだけの十分な情報を持たない場合はＰＨＴの内容に上書きしない。

【００７２】この発明の他の実施の形態も考えられる。例えば、特にタスク状態セグメントを持たないアーキテクチャでは、タスク切替えのときにポインタを書き換えて、ＰＨＴと同等のもの、即ち分岐パターン履歴に基づく予測情報、が与えられるメモリ内の位置を指すようにしてよい。この例では、別のタスクはその疑似ＰＨＴとは別の位置を持ち、タスク切替えのときに単にポインタを書き換える。この方法は、オペレーティングシステムによる命令制御の下で処理するのが最もよい。

【００７３】更に別の方法は、上に述べたように１つ以上のＰＨＴをＢＴＢ内に設けて、機械状態レジスタ（ＭＳＲ）への読取り及び書込みと同様に、プログラム制御の下でロード及び記憶の操作に用いることである。この方法はオペレーティングシステムによる命令制御の下でも同様に実行できる。

【００７４】更に、任意の上述の実施の形態及びその代替と組み合わせて、タスク切替えのときに分岐パターン

履歴に基づく予測情報の記憶やロードを選択的に可能にまた不可能にすることができる。例えば、ＭＳＲに可能／不可能ビットを設けて、その状態により予測情報の記憶及びロード動作が可能か不可能かを示す。この可能／不可能ビットのセットとリセットはプログラム制御の下で行う。

【００７５】この発明について好ましい実施の形態を参照して説明したが、当業者はこの明細書と図面を参照することにより、この発明の特徴と利点を実現するこれらの実施の形態の修正や代替を考えることができる。このような修正や代替はこの発明の特許請求の範囲内にあるものである。

【００７６】以上の説明に関して更に以下の項を開示する。

（１）　多重タスクモードで動作するマイクロプロセッサであって、第１及び第２のタスクに従って命令を実行する少なくとも１つの実行ユニットと、前記第１及び第２タスクのそれぞれに関連する部分を含み、また命令を記憶する部分を含む、メモリと、メモリにアドレスして前記実行ユニットが実行する命令コードを検索する取出しユニットであって、前記実行ユニットが実行した分岐命令の一連の結果を記憶する分岐履歴回路と、前記分岐履歴回路に結合し、前記分岐履歴回路からの分岐履歴フィールドに対応する予測情報を与える、パターン履歴回路と、取り出す命令のアドレスを選択する、アドレス指定回路と、を備える取出しユニットと、前記パターン履歴回路と前記メモリに結合し、前記第１タスクから前記第２タスクへのタスク切替えに応じて前記予測情報を修正する、回路と、を備える、マイクロプロセッサ。

【００７７】（２）　前記メモリは前記第１及び第２タスクにそれぞれ関連する第１及び第２タスク制御構造を備え、前記パターン履歴回路は複数の指標付けされた予測コードエントリを持ち、また前記分岐履歴回路の分岐履歴フィールドに対応する予測コードエントリの１つの内容を出す出力を持ち、また前記修正回路は、パターン履歴回路からの予測コードエントリを前記メモリに送って前記第１タスク制御構造内に記憶し、また前記第１タスクから前記第２タスクへのタスク切替えに応じて、予測コードエントリを第２のタスク制御構造からパターン履歴回路に送る、回路、を備える、第１項記載のマイクロプロセッサ。

【００７８】（３）　前記少なくとも１つの実行ユニットは前記メモリとの間でデータのロードと記憶を行うロード・記憶ユニットを備え、また前記通信回路は前記パターン履歴回路と前記ロード・記憶ユニットに結合するバスを備える、第２項記載のマイクロプロセッサ。

（４）　前記分岐履歴回路は、複数のエントリを有し、各エントリは関連する分岐命令の命令アドレスに対応するタグフィールドを持ち、またその関連する分岐命令の一連の前の分岐を記憶する分岐命令フィールドを持つ、分岐ターゲットバッファ、を備える、第２項記載のマイクロプロセッサ。

（５）　前記分岐ターゲットバッファ内の複数のエントリは分岐ターゲットアドレスを記憶するターゲットフィールドをそれぞれ有し、また前記アドレス指定回路は、分岐を行う予測に対応する出力を前記選択論理が出すとこれに応じて、前記関連する分岐命令に対応するエントリの分岐ターゲットアドレスに対応するアドレスを選択する、第４項記載のマイクロプロセッサ。

【００７９】（６）　前記パターン履歴回路は、前記分岐履歴回路に結合し、それぞれ複数の指標付けされた予測コードエントリを持ち、またそれぞれ前記分岐履歴回路からの分岐履歴フィールドに対応する予測コードエントリの１つの内容を出す出力を持つ、複数のパターン履歴表、を備え、前記実行ユニットは複数のプログラムの種類に従って命令を実行し、また前記取出しユニットは、プログラム種類標識を受けるように結合し、前記プログラム種類標識に対応する前記複数のパターン履歴表の１つの出力を前記アドレス指定回路に選択的に送る、選択論理、を更に有する、第２項記載のマイクロプロセッサ。

【００８０】（７）　分岐命令のプログラム種類標識は前記分岐命令を含むプログラムに対応する特権レベル標識を備える、第６項記載のマイクロプロセッサ。

（８）　前記特権レベル標識は前記分岐命令に対応する１ビットのコードセグメントレジスタを備える、第７項記載のマイクロプロセッサ。

（９）　分岐命令のプログラム種類標識は、前記分岐命令を含むメモリの一部として少なくとも１ビットのページテーブルエントリを有する、第６項記載のマイクロプロセッサ。

【００８１】（１０）　前記タスク制御構造はそれぞれ、タスク切替えに応じて前記パターン履歴回路からの予測コードを前記タスク制御構造からロードするかどうかを示す状態を記憶する、動的ロード制御ビットと、前記パターン履歴回路からの予測コードを記憶するメモリ位置、を有する、第２項記載のマイクロプロセッサ。

【００８２】（１１）　前記パターン履歴回路は、前記分岐履歴回路に結合し、それぞれ複数の指標付けされた予測コードエントリを持ち、またそれぞれ前記分岐履歴回路からの分岐履歴フィールドに対応する予測コードエントリの１つの内容を出す出力を持つ、複数のパターン履歴表、を備え、前記実行ユニットは複数のプログラムの種類に従って命令を実行し、前記取出しユニットは、プログラム種類標識を受けるように結合し、前記プログラム種類標識に対応する前記複数のパターン履歴表の１つの出力を前記アドレス指定回路に選択的に送る、選択論理、を更に備え、また前記タスク制御構造はそれぞれ、タスク切替えに応じて前記複数のパターン履歴表の関連する１つの予測コードを前記タスク制御構造からロ

ードするかどうかを示す状態を記憶する、複数の動的ロード制御ビットと、前記複数のパターン履歴表の関連する１つからの予測コードをそれぞれ記憶する、複数のメモリ位置と、を備える、第２項記載のマイクロプロセッサ。

【００８３】（１２）　前記メモリは、前記マイクロプロセッサの外部にある主メモリと、前記マイクロプロセッサと共にオンチップであるキャッシュメモリと、を備え、前記タスク制御構造は主メモリに記憶され、また前記キャッシュメモリは前記タスク制御構造の写しを含む、第２項記載のマイクロプロセッサ。

（１３）　パイプライン化マルチタスクのマイクロプロセッサを操作する方法であって、パイプライン化マイクロプロセッサの取出し段階で第１のタスクの分岐命令を検出し、前記検出ステップに応じて、分岐履歴フィールドの少なくとも一部を検索し、前記分岐履歴フィールドの前記検索された部分に対応する記憶された予測情報から、分岐予測を生成し、前記第１タスクから第２のタスクへのタスク切替えに応じて、前記予測情報を修正する、パイプライン化マルチタスクマイクロプロセッサを操作する方法。

【００８４】（１４）　分岐予測を生成する前記ステップは、前記検出された分岐命令の分岐履歴フィールドの検索された部分に対応する、パターン履歴表内に記憶されている予測情報を検索することを含み、前記修正するステップは、前記パターン履歴表からの予測情報を前記第１タスクに関連するメモリの第１のタスク制御構造部分に記憶し、前記第２タスクに関連するメモリの第２のタスク制御構造部分から予測情報をロードする、第１３項記載のパイプライン化マルチタスクマイクロプロセッサを操作する方法。

【００８５】（１５）　タスク切替えを行う前記ステップは、メモリの前記第１タスク制御構造部分内の動的ロードビットを調べる、ことを更に含み、また前記記憶するステップは、メモリの前記第１タスク制御構造部分内の動的ロードビットが予測情報をメモリの前記第１タスク制御構造部分に記憶すべきことを示すと、これにに応じて行う、第１４項記載のパイプライン化マルチタスクマイクロプロセッサを操作する方法。

（１６）　タスク切替えを行う前記ステップは、メモリの前記第２タスク制御構造部分内の動的ロードビットを調べる、ことを更に含み、また前記ロードするステップは、メモリの前記第２タスク制御構造部分内の動的ロードビットが、予測情報がメモリの前記第２タスク制御構造部分にあることを示すと、これに応じて行う、第１５項記載のパイプライン化マルチタスクマイクロプロセッサを操作する方法。

【００８６】（１７）　分岐予測を生成する前記ステップは、複数のパターン履歴表の中の選択された１つから検索された分岐履歴フィールドの部分に対応して行い、タスク切替えを行う前記ステップは、メモリの前記第１及び第２タスク制御構造部分内の、それぞれ前記複数のパターン履歴表の１つに関連する複数の動的ロードビットを調べる、ことを更に含み、前記記憶するステップは、メモリの前記第１タスク制御構造部分内の１つ以上の動的ロードビットが、前記複数のパターン履歴表の関連するものについて予測情報をメモリの前記第１タスク制御構造部分に記憶すべきことを示すと、これに応じて行い、また前記ロードするステップは、メモリの前記第２タスク制御構造部分内の１つ以上の動的ロードビットが、前記複数のパターン履歴表の関連するものについて予測情報がメモリの前記第２タスク制御構造部分にあることを示すと、これに応じて行う、第１４項記載のパイプライン化マルチタスクマイクロプロセッサを操作する方法。

【００８７】（１８）　前記分岐命令に対応するプログラムの種類を決定することを更に含み、また分岐予測を生成する前記ステップは、前記決定ステップで決定された前記プログラムの種類に従って選択された、複数のパターン履歴表の１つから検索された前記分岐履歴フィールドの部分に対応して行う、第１４項記載のパイプライン化マルチタスクマイクロプロセッサを操作する方法。

（１９）　前記決定するステップは、前記検出された分岐命令を含むプログラムに対応する特権レベル標識の状態を調べる、ことを含む、第１８項記載のパイプライン化マルチタスクマイクロプロセッサを操作する方法。

【００８８】（２０）　前記特権レベル標識は、前記検出された分岐命令に対応する１ビットのコードセグメントレジスタを備える、第１９項記載のパイプライン化マルチタスクマイクロプロセッサを操作する方法。

（２１）　前記決定するステップは、前記検出された分岐命令を含むメモリの一部について少なくとも１ビットのページテーブルエントリの状態を調べる、ことを含む、第１８項記載のパイプライン化マルチタスクマイクロプロセッサを操作する方法。

【００８９】（２２）　前記第１及び第２タスクの予測情報はメモリの第１及び第２部分にそれぞれ記憶され、前記生成するステップは、前記分岐履歴フィールドの検索された部分に対応するメモリの前記第１部分から予測情報を検索することを含み、前記修正するステップは、メモリの前記第２部分を指すようポインタを書き直し、前記第２タスク内の分岐命令を検出すると、前記生成するステップはメモリの前記第２部分から予測情報を検索するようにする、ことを含む、第１３項記載のパイプライン化マルチタスクマイクロプロセッサを操作する方法。

【００９０】（２３）　マイクロプロセッサとこれを含むシステムであって、分岐命令を含むプログラムの種類に応じて分岐予測を行う。取出しユニット（２６）は分岐ターゲットバッファ（５６）と、複数のパターン履歴

表（５３）を有する。選択論理（８０）は各分岐命令毎に、命令を含むプログラムの種類を示す信号を受けて１つのパターン履歴表（５３）を選択し、これを用いて、命令アドレスに対応する分岐ターゲットバッファ（５６）のエントリ内の分岐履歴フィールド（ＢＨ）の一部に応じて予測コードを生成する。パターン履歴表（５３）を選択するのに用いる信号の例として、命令の特権レベル（即ち、ユーザレベルかスーパーバイザレベル）の指標（Ｕ／Ｓ）がある。タスク切替えの場合は、１つ以上のパターン履歴表（５３）の内容を、中断されたタスクに対応するタスク状態セグメント（９０）に記憶し、新しいタスクのタスク状態セグメントからのエントリをパターン履歴表（５３）にロードする。このようにして、マイクロプロセッサをマルチタスク環境で動作させるとき、各タスクは自身の分岐パターン履歴に基づく予測情報を保持する。

【図面の簡単な説明】

【図１】この発明の好ましい実施の形態のマイクロプロセッサとシステムのブロック図。

【図２】この発明の好ましい実施の形態の図１のマイクロプロセッサ内の取出しユニットのブロック図。

【図３】この発明の好ましい実施の形態の図１のマイクロプロセッサ内の分岐ターゲットバッファとパターン履歴表と関連する回路のブロック図。

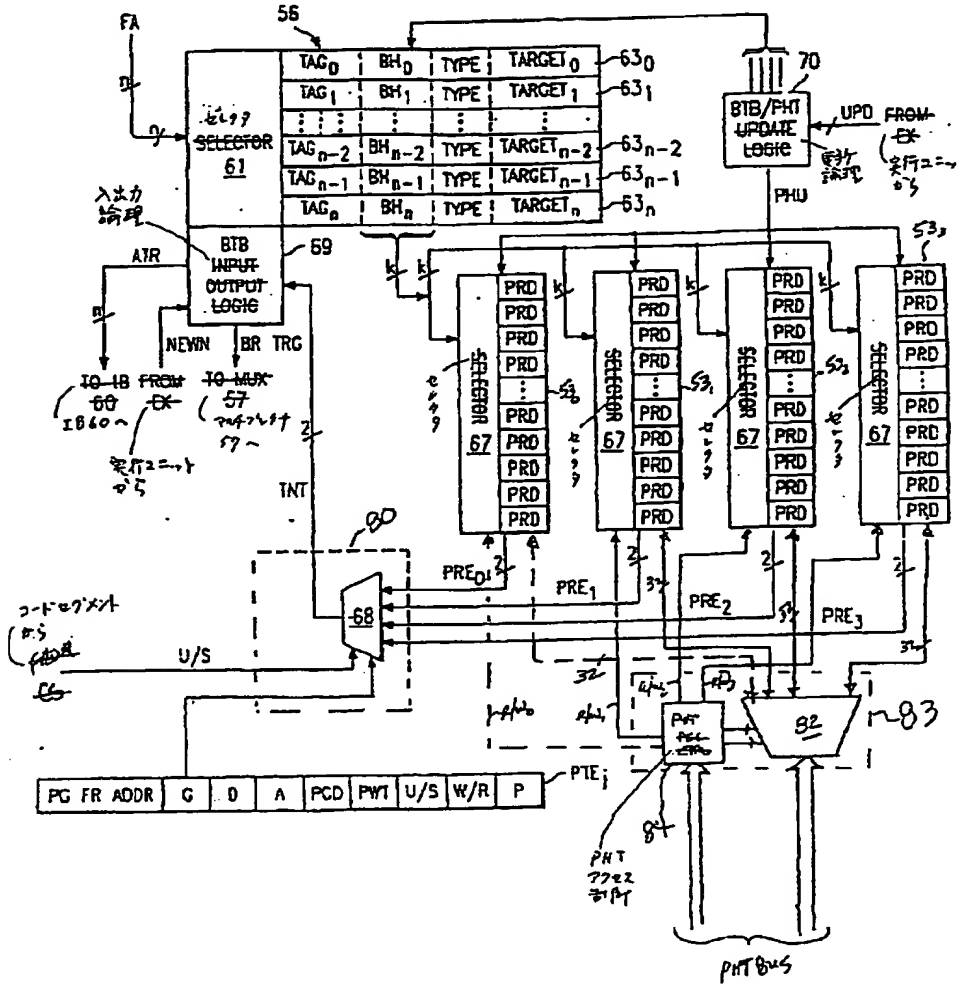
【図４】この発明の好ましい実施の形態のタスク状態セグメントの配置を示すメモリマップ。

【図５】この発明の好ましい実施の形態のタスク切替えルーチンの一部を示す流れ図。

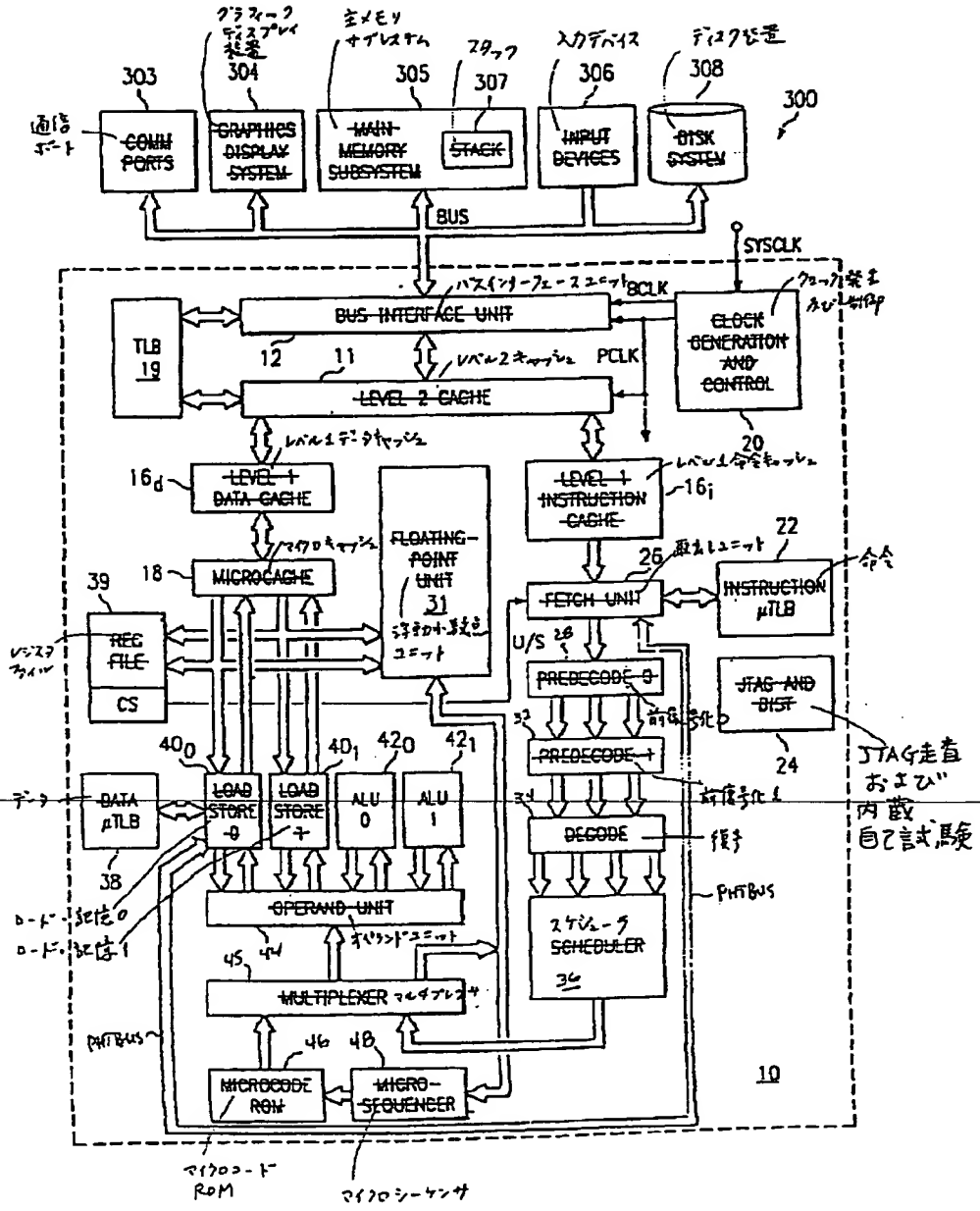
【符号の説明】

２６　　取出しユニット
５３　　パターン履歴表
５６　　分岐ターゲットバッファ
８０　　選択論理
９０　　タスク状態セグメント

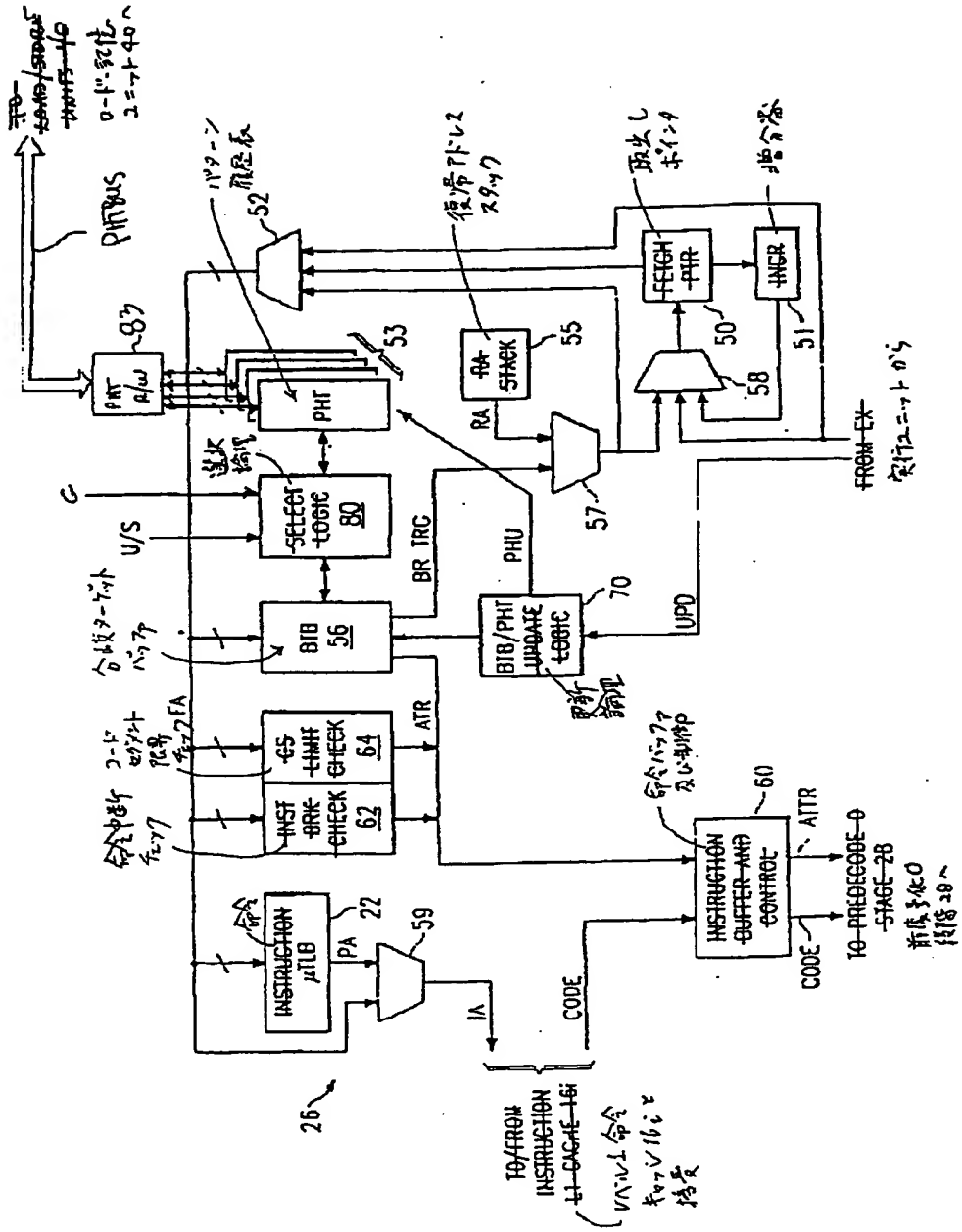
【図３】



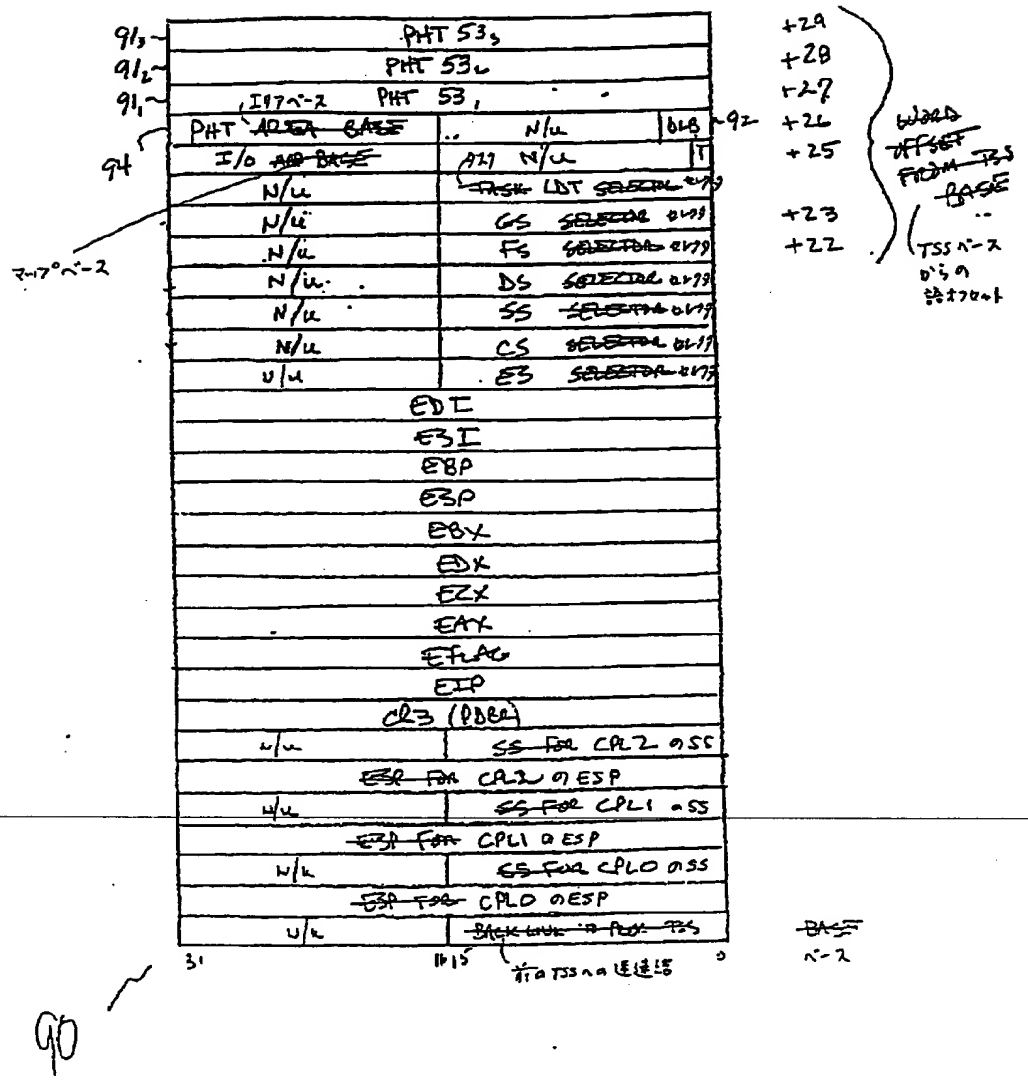
【図１】


303
304
305
307
306
308
300
COMM PORTS
GRAPHICS DISPLAY SYSTEM
MAIN MEMORY SUBSYSTEM
STACK
INPUT DEVICES
DISK SYSTEM
BUS
SYSCLK
BUS INTERFACE UNIT
BCLK
CLOCK GENERATION AND CONTROL
TLB 19
12
11
LEVEL 2 CACHE
PCLK
20
LEVEL 1 DATA CACHE
16d
LEVEL 1 INSTRUCTION CACHE
16i
FLOATING-POINT UNIT 31
MICROCACHE
18
39
22
26
FETCH UNIT
INSTRUCTION μTLB
REG FILE
CS
U/S 28
PREDECODE 0
PREDECODE 1
32
34
JTAG AND BIST
24
DECODE
DATA μTLB
38
LOAD STORE 0
LOAD STORE 1
ALU 0
ALU 1
40₀
40₁
42₀
42₁
OPERAND UNIT
45
44
MULTIPLEXER
PHYBUS
46
48
MICROCODE ROM
MICRO-SEQUENCER
SCHEDULER 36
10
JTAG走査 および 内蔵 自己試験
マイクロコードROM
マイクロシーケンサ

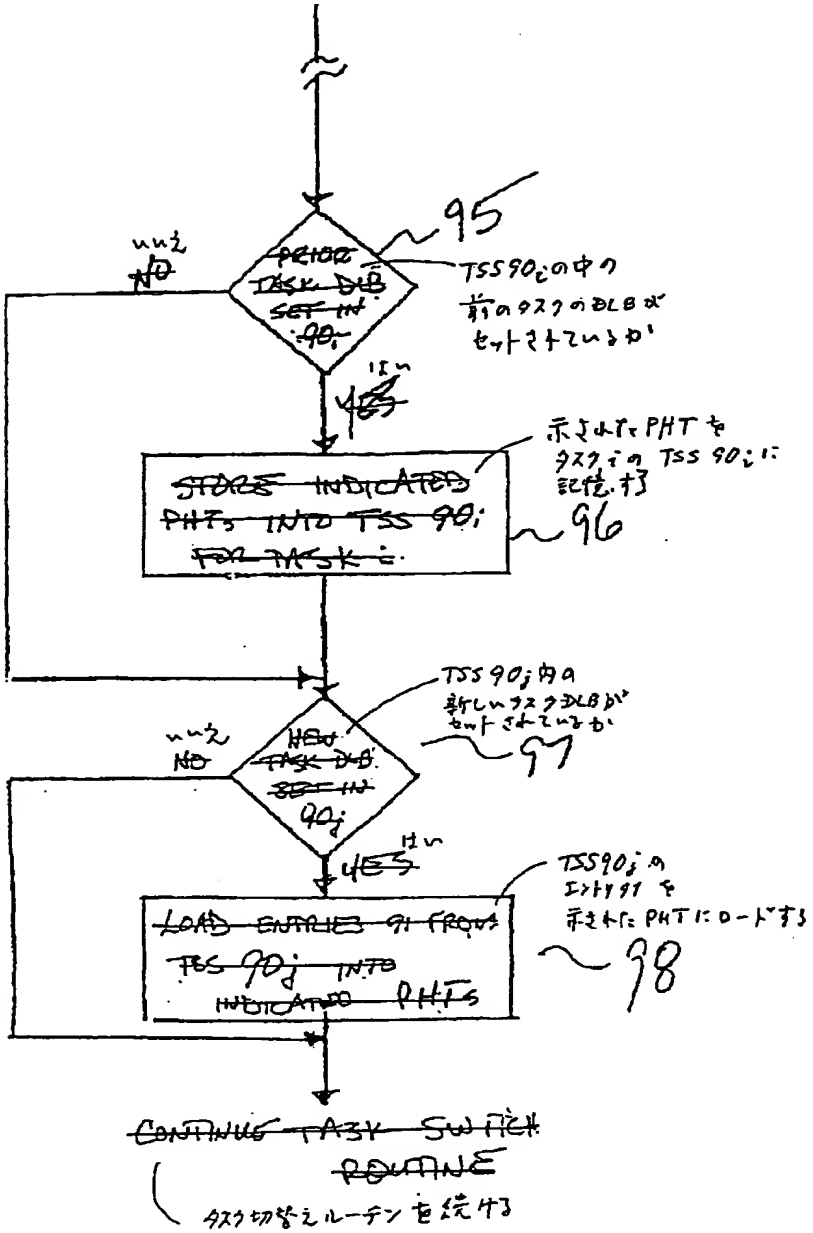
【図2】


PHYBUS
PHT
PHT R/W
SELECT LOGIC 80
BIB 56
BIB/PHT UPDATE LOGIC
70
BR TRC
PHU
UPD
RA STACK
55
FETCH ADR
50
INSTR
51
58
57
FROM EX
52
53
U/S
C
INSTRUCTION μTLB
22
PA
59
IA
CODE
26
INSTRUCTION BUFFER AND CONTROL
60
ATTR
CODE
TO PREDECODE-0 STAGE 28
TO/FROM INSTRUCTION L1 CACHE 16
INST BRK CHECK 62
65 HALF CHECK 64
ATR

【図4】


91₃
91₂
91₁
94
92
PHT 53₃
PHT 53₂
I/Oベース PHT 53₁
PHT ~~AREA BASE~~
N/U
BLB
I/O ~~MAP BASE~~
N/U
T
~~TASK~~ LDT SELECTOR
GS SELECTOR
FS SELECTOR
DS SELECTOR
SS SELECTOR
CS SELECTOR
ES SELECTOR
EDI
ESI
EBP
ESP
EBX
EDX
ECX
EAX
EFLAG
EIP
CR3 (PDBR)
SS FOR CPL2 のSS
ESP FOR CPL2 のESP
SS FOR CPL1 のSS
ESP FOR CPL1 のESP
SS FOR CPL0 のSS
ESP FOR CPL0 のESP
~~BACK LINK TO PREV TSS~~
31
15
0
前のTSSへの連結部
マップベース
+29
+28
+27
+26
+25
+23
+22
WORD OFFSET FROM TSS ~~BASE~~
TSSベースからのオフセット
~~BASE~~ ベース
90

【図５】


いいえ
NO
95
TSS 90$_i$の中の
前のタスクのDLBが
セットされているか
はい
YES
示されたPHTを
タスクiのTSS 90$_i$に
記憶する
96
TSS 90$_j$内の
新しいタスクのDLBが
セットされているか
97
いいえ
NO
はい
YES
TSS90$_j$の
エントリを
示されたPHTにロードする
98
タスク切替えルーチンを続行